Appendix: Mathematical and Computational Background

If you wish to learn about nature, to appreciate nature, it is necessary to understand the language she speaks in. She offers her information only in one form; we are not so unhumble as to demand that she change before we pay attention.

Richard Feynman, *The Character of Physical Law* (1982)

A.1 Introduction

The study of seismology follows a pattern characteristic of many scientific disciplines. We first identify phenomena that we seek to understand, such as the propagation of seismic waves through the solid earth. We then consider the physics of the simplest relevant case, such as the propagation of a wave of a single frequency through a uniform material, formulate the problem mathematically, and derive a solution. From this solution, we build up mathematical solutions to more complex problems, each of which is ideally a better approximation to the complexities of the real earth. Although the simpler problems can be solved analytically, eventually the complexities require numerical techniques.

We thus rely on a set of mathematical techniques often used in physical problems. Experience suggests that although many readers are familiar with most of the mathematics required in this book, a review is often helpful. This appendix briefly summarizes a broad range of material. The first sections treat a variety of mathematical topics. The final section reviews some concepts relevant to the use of computers for scientific calculations.

In using these mathematical techniques, it is worth bearing in mind that we are invoking the special power of mathematics to deal with physical problems. This power is that if a physical problem is posed correctly in mathematical terms, then applying mathematical techniques to this formulation yields quite different, and often apparently unrelated, statements that also correctly describe the physical world. For example, in Section 2.4 we used the equations of elasticity and applied vector calculus to derive the properties of seismic waves that we observe. Similarly, in Section 2.5 we derived an observed physical relation, Snell's law, starting from three different physical formulations. Conversely, we have seen that different physical phenomena can be described using similar mathematical approaches and so have some deep similarities. Although in hindsight such successes may not seem surprising, because many of the mathematical methods we use were developed to solve such physical problems, they illustrate the intimate connection between sciences like seismology and mathematics.[1]

A.2 Complex numbers

In several of our applications, notably in describing propagating waves and their frequency content, complex numbers are helpful. We thus briefly review some of their properties.

The complex number $z = a + ib$, where $i = \sqrt{-1}$, has a real part, a, and an imaginary part, b. These relations are sometimes written $a = \mathrm{Re}\,(z)$ and $b = \mathrm{Im}\,(z)$. Complex numbers are typically plotted in the complex plane with their real parts on the x_1 axis and their imaginary parts on the x_2 axis (Fig. A.2-1). Alternatively, a complex number can be written in *polar coordinate* form as

$$z = a + ib = re^{i\theta} = r(\cos\theta + i\sin\theta). \tag{1}$$

[1] Most seismologists are more conservative than Paul Dirac, a leader in the development of quantum physics, who invented the delta function. Dirac regarded mathematical beauty as a guiding principle, stating that "it is more important to have beauty in one's equations than to have them fit experiment."

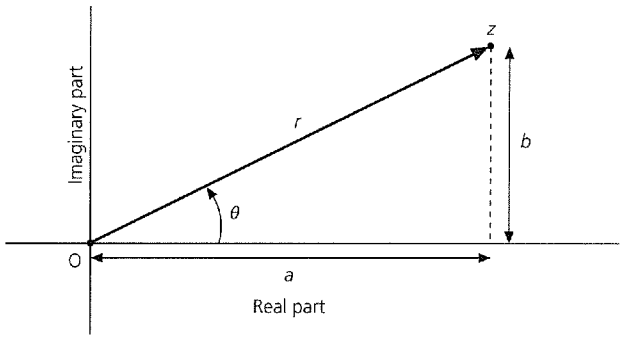


Fig. A.2-1 A number in the complex plane can be represented in terms of its real and imaginary parts, $z = a + ib$, or in polar form $z = re^{i\theta}$.

The *polar coordinates*, the magnitude r and the phase angle θ, can be expressed in terms of the real and imaginary parts as

$$r = \sqrt{a^2 + b^2}, \quad \theta = \tan^{-1}(b/a). \tag{2}$$

and, conversely,

$$a = r\cos\theta, \quad b = r\sin\theta. \tag{3}$$

To describe complex numbers in all four quadrants of the complex plane, θ ranges from 0 to 2π. Because the inverse tangent is periodic with period π, the signs of the real and imaginary parts are used to obtain the correct phase.

Complex numbers are equal when they have the same real and imaginary parts. Two complex numbers in $(a + ib)$ form are added by adding the real parts and the imaginary parts:

$$(a_1 + ib_1) + (a_2 + ib_2) = (a_1 + a_2) + i(b_1 + b_2). \tag{4}$$

Complex numbers can be multiplied either in the $(a + ib)$ form:

$$(a_1 + ib_1)(a_2 + ib_2) = (a_1a_2 - b_1b_2) + i(a_1b_2 + b_1a_2), \tag{5}$$

or in the magnitude and phase form:

$$r_1e^{i\theta_1}r_2e^{i\theta_2} = r_1r_2e^{i(\theta_1+\theta_2)}. \tag{6}$$

The conjugate of a complex number z, z^*, has the same real part and an imaginary part of opposite sign. Because

$$\begin{aligned} z^* &= a - ib = r\cos\theta - ir\sin\theta \\ &= r\cos(-\theta) + ir\sin(-\theta) = re^{-i\theta}, \end{aligned} \tag{7}$$

the conjugate has the same magnitude but the opposite phase. Hence the square of the magnitude of a complex number can be found by multiplication by the complex conjugate,

$$|z|^2 = zz^* = (a + ib)(a - ib) = (a^2 + b^2) = re^{i\theta}re^{-i\theta} = r^2. \tag{8}$$

By combining

$$e^{i\theta} = \cos\theta + i\sin\theta \quad \text{and} \quad e^{-i\theta} = \cos\theta - i\sin\theta \tag{9}$$

we obtain the definitions of the sine and cosine functions in terms of complex exponentials

$$\cos\theta = (e^{i\theta} + e^{-i\theta})/2 \quad \text{and} \quad \sin\theta = (e^{i\theta} - e^{-i\theta})/2i. \tag{10}$$

These relations yield formulae for the trigonometric functions of the sum of the angles because

$$e^{i(\theta_1+\theta_2)} = \cos(\theta_1 + \theta_2) + i\sin(\theta_1 + \theta_2) \tag{11}$$

and, by Eqn 6,

$$\begin{aligned} e^{i(\theta_1+\theta_2)} = e^{i\theta_1}e^{i\theta_2} &= (\cos\theta_1 + i\sin\theta_1)(\cos\theta_2 + i\sin\theta_2) \\ &= (\cos\theta_1\cos\theta_2 - \sin\theta_1\sin\theta_2) \\ &\quad + i(\sin\theta_1\cos\theta_2 + \cos\theta_1\sin\theta_2), \end{aligned} \tag{12}$$

so we can equate the real and imaginary parts and find

$$\cos(\theta_1 + \theta_2) = \cos\theta_1\cos\theta_2 - \sin\theta_1\sin\theta_2 \tag{13}$$

and

$$\sin(\theta_1 + \theta_2) = \sin\theta_1\cos\theta_2 + \cos\theta_1\sin\theta_2. \tag{14}$$

These expressions are symmetric in θ_1 and θ_2, as expected. The corresponding relations for the trigonometric functions of the difference of two angles are found by making θ_2 negative. Setting $\theta_1 = \theta_2$ gives expressions for $\cos(2\theta)$ and $\sin(2\theta)$.

The relations for the product of trigonometric functions of two angles can also be found using complex exponentials

$$\begin{aligned} \cos\theta_1\cos\theta_2 &= \frac{(e^{i\theta_1} + e^{-i\theta_1})}{2}\frac{(e^{i\theta_2} + e^{-i\theta_2})}{2} \\ &= \frac{1}{4}[(e^{i(\theta_1+\theta_2)} + e^{-i(\theta_1+\theta_2)}) + (e^{i(\theta_1-\theta_2)} + e^{-i(\theta_1-\theta_2)})] \\ &= \frac{1}{2}[\cos(\theta_1 + \theta_2) + \cos(\theta_1 - \theta_2)] \end{aligned} \tag{15}$$

and, similarly,

$$\begin{aligned} \sin\theta_1\sin\theta_2 &= \frac{(e^{i\theta_1} - e^{-i\theta_1})}{2i}\frac{(e^{i\theta_2} - e^{-i\theta_2})}{2i} \\ &= \frac{1}{4}[(e^{i(\theta_1-\theta_2)} + e^{-i(\theta_1-\theta_2)}) - (e^{i(\theta_1+\theta_2)} + e^{-i(\theta_1+\theta_2)})] \\ &= \frac{1}{2}[\cos(\theta_1 - \theta_2) - \cos(\theta_1 + \theta_2)]. \end{aligned} \tag{16}$$

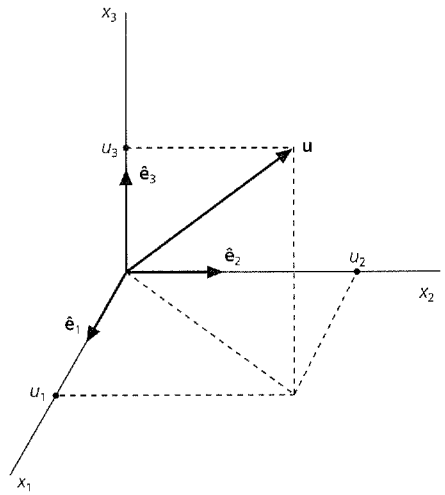


Fig. A.3-1 A vector **u** is expressed by the Cartesian unit basis vectors and its components: $\mathbf{u} = u_1\hat{e}_1 + u_2\hat{e}_2 + u_3\hat{e}_3$.

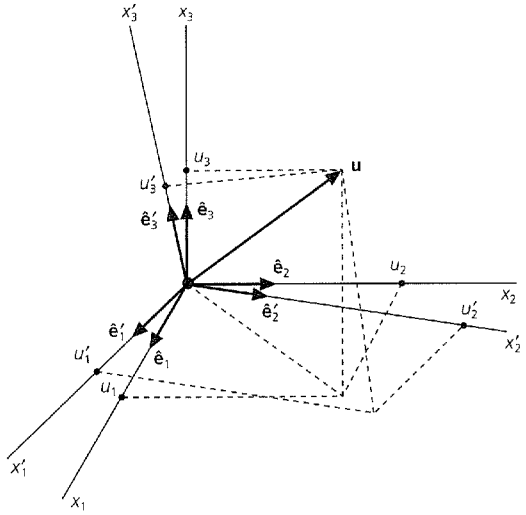


Fig. A.3-2 A vector **u** is described in each of two orthogonal coordinate systems by the Cartesian unit basis vectors of the coordinate system and the components of the vector in the coordinate system: $\mathbf{u} = u_1\hat{e}_1 + u_2\hat{e}_2 + u_3\hat{e}_3 = u_1'\hat{e}_1' + u_2'\hat{e}_2' + u_3'\hat{e}_3'$. Although the components differ between coordinate systems, the vector remains the same.

A.3 Scalars and vectors

A.3.1 Definitions

In seismology, we deal with several types of physical quantities. The simplest, *scalars*, are numbers describing a physical property at a given point that is independent of the coordinate system used to identify the point. Temperature, pressure, mass, and density are familiar examples. Mathematically, if a point is described in one coordinate system by (x_1, x_2, x_3) and in a second by (x_1', x_2', x_3'), the value of a scalar function ϕ in the first coordinate system equals that of the corresponding scalar function in the second

$$\phi(x_1, x_2, x_3) = \phi'(x_1', x_2', x_3'). \tag{1}$$

The distance between two points is a scalar because although the coordinates of the points depend on the coordinate system, the distance does not.

Vectors are more complicated entities that have magnitude and direction. In seismology, the most common vector is the motion, or *displacement*, of a piece of material within the earth due to the passage of a seismic wave. Vectors transform between different coordinate systems in a specific way. Thus, if the horizontal ground motion is recorded with seismometers oriented northeast–southwest and northwest–southeast, the north–south and east–west components of the displacement can be found using the properties of vectors. We will see that although the components depend on the coordinate system, the magnitude and direction of the vector remain the same.

Consider the familiar Cartesian coordinate system (Fig. A.3-1) with three mutually perpendicular (orthogonal) coordinate axes. There are two standard notations for these coordinates and axes: either the x_1, x_2, and x_3, or the x, y, and z axes. Each notation has advantages. The x_1, x_2, x_3 notation is more convenient for some derivations, and the x, y, z notation is sometimes clearer in physical problems. We use the x_1, x_2, and x_3 notation in this appendix, and use whichever notation seems more convenient in other discussions.

A point in this coordinate system is described by its x_1, x_2, and x_3 coordinates. Because a vector can be defined by a line from the origin (0, 0, 0) to the point (u_1, u_2, u_3), the three numbers u_1, u_2, and u_3 are the *components* of the vector **u**. A vector is denoted either by boldface type or by a set of its components

$$\mathbf{u} = (u_1, u_2, u_3) = (u_x, u_y, u_z). \tag{2}$$

A Cartesian coordinate system is described by three orthogonal unit basis vectors, $\hat{e}_1$, $\hat{e}_2$, and $\hat{e}_3$, along the x_1, x_2, and x_3 coordinate axes:

$$\hat{e}_1 = (1, 0, 0) \quad \hat{e}_2 = (0, 1, 0) \quad \hat{e}_3 = (0, 0, 1). \tag{3}$$

The caret, or "hat" superscript, indicates a *unit vector*, whose length is 1. The vector **u** is formed from its components and the basis vectors

$$\mathbf{u} = u_1\hat{e}_1 + u_2\hat{e}_2 + u_3\hat{e}_3 = (u_1, u_2, u_3). \tag{4}$$

Now, consider a second Cartesian coordinate system with the same origin and different axes x_1', x_2', and x_3', along which unit basis vectors $\hat{e}_1'$, $\hat{e}_2'$, and $\hat{e}_3'$ are defined (Fig. A.3-2). In this coordinate system the components of **u** are different,

$$\mathbf{u} = u_1'\hat{e}_1' + u_2'\hat{e}_2' + u_3'\hat{e}_3' = (u_1', u_2', u_3'). \tag{5}$$

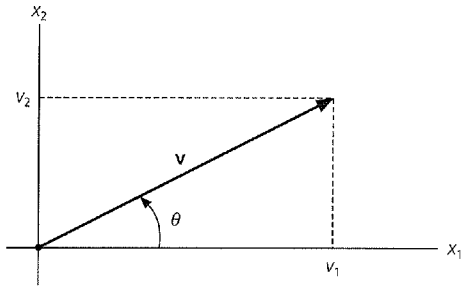


Fig. A.3-3 A vector in two dimensions making an angle θ with the x_1 axis.

Thus the *same* physical vector is represented in a different coordinate system, described by a different set of basis vectors, using different components. The essential idea is that the vector remains the same, or invariant, regardless of the coordinate system, although the numerical values of its components change. Physical laws, like Newton's law stating that the force vector equals the product of the mass and the acceleration vector (the second derivative with respect to time of the displacement vector), are written in vector form because the physical phenomenon does not depend on the coordinate system used to describe it.

The length or *magnitude* of a vector, $|\mathbf{u}|$, is a scalar, and thus the same in different coordinate systems. By the Pythagorean theorem, the length is

$$|\mathbf{u}| = (u_1^2 + u_2^2 + u_3^2)^{1/2} = (u_1'^2 + u_2'^2 + u_3'^2)^{1/2}. \tag{6}$$

The zero vector, 0, all of whose components are zero in any coordinate system, has zero magnitude.

A vector is specified in either Cartesian coordinates by its components or in polar coordinates by its magnitude and direction. For example, in a two-dimensional (x_1, x_2) coordinate system (Fig. A.3-3), the vector $\mathbf{v}$ can be written in terms of its components

$$\mathbf{v} = (v_1, v_2) \tag{7}$$

or its magnitude

$$|\mathbf{v}| = (v_1^2 + v_2^2)^{1/2} \tag{8}$$

and direction, given by the angle θ that $\mathbf{v}$ makes with the x_1 direction

$$\theta = \tan^{-1}(v_2/v_1). \tag{9}$$

Just as $|\mathbf{v}|$ and θ are given by the components, so the components are given by $|\mathbf{v}|$ and θ

$$v_1 = |\mathbf{v}| \cos\theta \quad \text{and} \quad v_2 = |\mathbf{v}| \sin\theta. \tag{10}$$

By analogy, a vector in three dimensions is specified by either its three components or its magnitude and the angles it forms with two of the coordinate axes. It is worth noting that the mathematical convention of defining angles counterclockwise from x_1 differs from the geographical convention of defining angles clockwise from North (x_2), so conversions are often needed.

A.3.2 Elementary vector operations

The simplest vector operation is multiplication of a vector by a scalar

$$\alpha\mathbf{u} = (\alpha u_1, \alpha u_2, \alpha u_3). \tag{11}$$

For example, in two dimensions,

$$\alpha\mathbf{v} = (\alpha v_1, \alpha v_2) \tag{12}$$

yields a vector with magnitude

$$((\alpha v_1)^2 + (\alpha v_2)^2)^{1/2} = |\alpha|\,(v_1^2 + v_2^2)^{1/2} = |\alpha|\,|\mathbf{v}| \tag{13}$$

whose direction is given by

$$\tan\theta = \alpha v_2/\alpha v_1 = v_2/v_1. \tag{14}$$

Multiplication by a positive scalar thus changes the magnitude of a vector but preserves its direction. Similarly, multiplication by a negative scalar changes the magnitude of a vector and reverses its direction. $\hat{\mathbf{u}}$, a unit vector in the direction of $\mathbf{u}$ is formed by dividing $\mathbf{u}$ by its magnitude

$$\hat{\mathbf{u}} = \mathbf{u}/|\mathbf{u}|. \tag{15}$$

The sum of two vectors is another vector whose components are the sums of the corresponding components, so if

$$\mathbf{a} = a_1\hat{\mathbf{e}}_1 + a_2\hat{\mathbf{e}}_2 + a_3\hat{\mathbf{e}}_3 \quad \text{and} \quad \mathbf{b} = b_1\hat{\mathbf{e}}_1 + b_2\hat{\mathbf{e}}_2 + b_3\hat{\mathbf{e}}_3,$$

$$\mathbf{a} + \mathbf{b} = (a_1 + b_1)\hat{\mathbf{e}}_1 + (a_2 + b_2)\hat{\mathbf{e}}_2 + (a_3 + b_3)\hat{\mathbf{e}}_3 = \mathbf{b} + \mathbf{a}. \tag{16}$$

Addition can be done graphically (Fig. A.3-4) by shifting one vector, while preserving its orientation, so that its "tail" is at the "head" of the other, and forming the vector sum. For example, the total force vector acting on an object is the vector sum of the individual force vectors. Equation 16 and Fig. A.3-4 show that vector addition is commutative; it does not matter in which order the vectors are added.

A.3.3 Scalar products

There are two methods of multiplying vectors. The first, the *scalar product* (also called the dot product or inner product), yields a scalar:

$$\mathbf{a} \cdot \mathbf{b} = a_1b_1 + a_2b_2 + a_3b_3 = |\mathbf{a}|\,|\mathbf{b}| \cos\theta, \tag{17}$$

where θ is the angle between two vectors. To see that the two definitions of the scalar product are equivalent, consider a two-

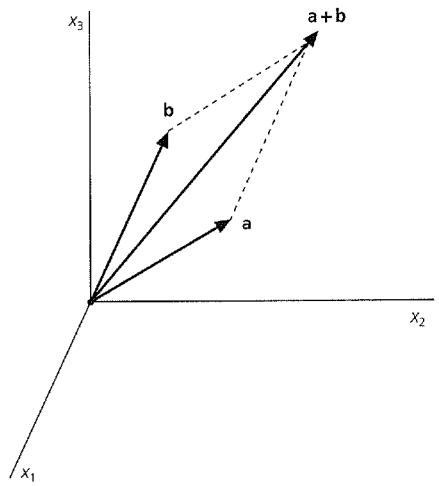


Fig. A.3-4 Addition of vectors **a** and **b**. The addition can be done analytically, by adding components, or graphically. Vector addition is commutative, as the order of addition is irrelevant.

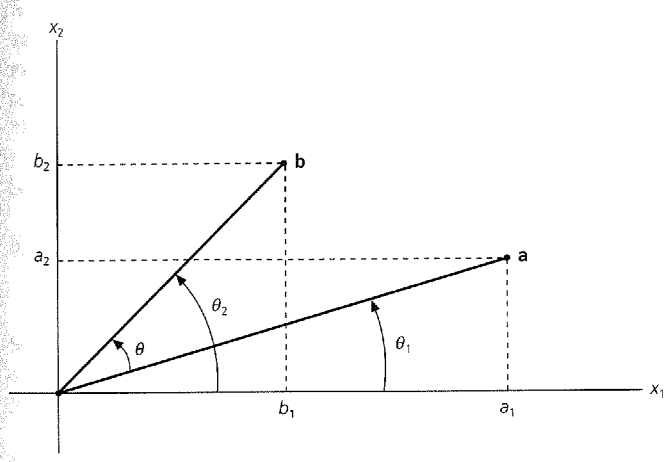


Fig. A.3-5 Derivation of alternative definitions of the scalar product $\mathbf{a} \cdot \mathbf{b}$ in two dimensions.

dimensional case (Fig. A.3-5) with $\mathbf{a} = (a_1, a_2)$ and $\mathbf{b} = (b_1, b_2)$. If **a** and **b** make angles θ_1 and θ_2 with the $\hat{\mathbf{e}}_1$ axis, then

$$\mathbf{a} \cdot \mathbf{b} = |\mathbf{a}||\mathbf{b}| \cos \theta = |\mathbf{a}||\mathbf{b}| \cos (\theta_2 - \theta_1). \tag{18}$$

Using a trigonometric identity (Eqn A.2.13) we expand

$$\cos \theta = \cos (\theta_2 - \theta_1) = \cos \theta_2 \cos \theta_1 + \sin \theta_2 \sin \theta_1. \tag{19}$$

Because

$$\cos \theta_1 = a_1/(a_1^2 + a_2^2)^{1/2} \quad \text{and} \quad \sin \theta_1 = a_2/(a_1^2 + a_2^2)^{1/2}, \tag{20}$$

and similar definitions hold for θ_2 and **b**, substitutions for the angles in Eqn 18 show that

$$|\mathbf{a}||\mathbf{b}| \cos \theta = \frac{|\mathbf{a}||\mathbf{b}|(a_1 b_1 + a_2 b_2)}{(a_1^2 + a_2^2)^{1/2}(b_1^2 + b_2^2)^{1/2}} = a_1 b_1 + a_2 b_2. \tag{21}$$

Equation 17 shows several features of the scalar product:

- The scalar product commutes: $\mathbf{a} \cdot \mathbf{b} = \mathbf{b} \cdot \mathbf{a}$.
- The scalar product of two perpendicular vectors is zero, because $\cos 90° = 0$.
- The scalar product of a vector with itself is its magnitude squared:

$$\mathbf{a} \cdot \mathbf{a} = a_1 a_1 + a_2 a_2 + a_3 a_3 = |\mathbf{a}|^2. \tag{22}$$

The definition of the scalar product is generalized for vectors with complex components. To see why, note that for a vector $\mathbf{a} = (i, 1, 0)$, where $i = \sqrt{-1}$, Eqn 22 would give a squared magnitude of zero. Because we would like only the zero vector, all of whose elements are zero, to have zero magnitude, Eqn 17 is generalized to

$$\mathbf{a} \cdot \mathbf{b} = a_1^* b_1 + a_2^* b_2 + a_3^* b_3 \tag{23}$$

where * indicates the complex conjugate. Thus the definition of the squared magnitude (Eqn 22) becomes

$$\mathbf{a} \cdot \mathbf{a} = a_1^* a_1 + a_2^* a_2 + a_3^* a_3 = |\mathbf{a}|^2. \tag{24}$$

For example, the squared magnitude of $|(i, 1, 0)|^2 = (i)(-i) + (1)(1) = 2$. These complex definitions reduce to the familiar cases, (Eqns 17 and 22), for vectors with real components.

The relations between the unit basis vectors for a Cartesian coordinate system, $\hat{\mathbf{e}}_1$, $\hat{\mathbf{e}}_2$, and $\hat{\mathbf{e}}_3$, are easily stated using their scalar products. Because each is perpendicular to the other two, the scalar product of any two different ones is zero,

$$\hat{\mathbf{e}}_1 \cdot \hat{\mathbf{e}}_2 = \hat{\mathbf{e}}_1 \cdot \hat{\mathbf{e}}_3 = \hat{\mathbf{e}}_2 \cdot \hat{\mathbf{e}}_3 = 0, \tag{25}$$

and the scalar product of each with itself is its squared magnitude

$$\hat{\mathbf{e}}_1 \cdot \hat{\mathbf{e}}_1 = \hat{\mathbf{e}}_2 \cdot \hat{\mathbf{e}}_2 = \hat{\mathbf{e}}_3 \cdot \hat{\mathbf{e}}_3 = 1. \tag{26}$$

The unit basis set of vectors is *orthonormal*; each is orthogonal (perpendicular) to the others and normalized to unit magnitude.

The *projection*, or component of a vector in a direction given by a unit vector, is the scalar product of a vector with the unit vector. Using this idea, a component of a vector can be found from its projection on the unit basis vector along the corresponding axis. Thus the x_1 component of **u** is

$$\mathbf{u} \cdot \hat{\mathbf{e}}_1 = (u_1 \hat{\mathbf{e}}_1 + u_2 \hat{\mathbf{e}}_2 + u_3 \hat{\mathbf{e}}_3) \cdot \hat{\mathbf{e}}_1 = u_1, \tag{27}$$

with the other components defined similarly.

A.3.4 Vector products

A second form of multiplication, the *vector* or *cross* product, forms a third vector from two vectors by

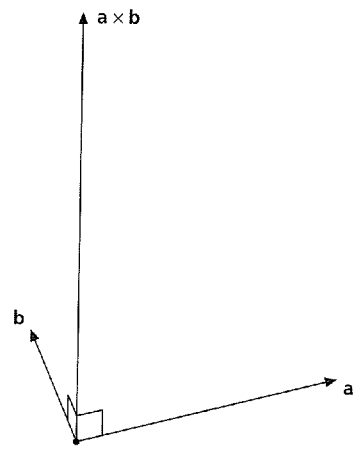


Fig. A.3-6 Illustration of the right-hand rule giving the orientation of the vector product $\mathbf{a} \times \mathbf{b}$.

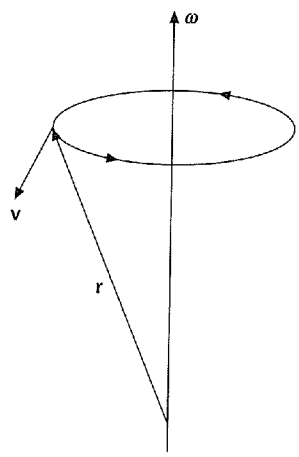


Fig. A.3-7 The vector product $\mathbf{v} = \boldsymbol{\omega} \times \mathbf{r}$ describes a rotation.

$$\mathbf{a} \times \mathbf{b} = (a_2 b_3 - a_3 b_2)\hat{\mathbf{e}}_1 + (a_3 b_1 - a_1 b_3)\hat{\mathbf{e}}_2 + (a_1 b_2 - a_2 b_1)\hat{\mathbf{e}}_3, \tag{28}$$

which can be written as the determinant

$$\mathbf{a} \times \mathbf{b} = \begin{vmatrix} \hat{\mathbf{e}}_1 & \hat{\mathbf{e}}_2 & \hat{\mathbf{e}}_3 \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{vmatrix}. \tag{29}$$

The vector product of two vectors is perpendicular to both vectors. For example, if **a** and **b** are in the x_1–x_2 plane, $a_3 = b_3 = 0$, and by Eqn 28, the vector product has only an $\hat{\mathbf{e}}_3$ component. This can be shown in general by evaluating $\mathbf{a} \cdot (\mathbf{a} \times \mathbf{b}) = \mathbf{b} \cdot (\mathbf{a} \times \mathbf{b}) = 0$. Geometrically, the direction of the vector product is found by a "right-hand rule" (Fig. A.3-6): if the fingers of a right hand rotate from **a** to **b**, the thumb points in the direction $\mathbf{a} \times \mathbf{b}$. The magnitude of the cross product is

$$|\mathbf{a} \times \mathbf{b}| = |\mathbf{a}|\,|\mathbf{b}| \sin \theta, \tag{30}$$

where θ is the angle between the two vectors. The cross product is zero for parallel vectors because $\sin 0° = 0$, so the cross product of a vector with itself is zero.

The vector product often appears in connection with rotations, such as those used to describe the motion of lithospheric plates (Section 5.2). For example, if an object located at a position **r** undergoes a rotation, its linear velocity **v** is given by

$$\mathbf{v} = \boldsymbol{\omega} \times \mathbf{r}, \tag{31}$$

where $\boldsymbol{\omega}$ is the rotation vector, which is oriented along the axis of rotation, with a magnitude $|\boldsymbol{\omega}|$ that is the angular velocity (Fig. A.3-7). Similarly, the vector product is used to define the torque, which gives the rate of change of angular momentum. A force **F**, acting at a point **r**, gives a torque

$$\boldsymbol{\tau} = \mathbf{r} \times \mathbf{F}. \tag{32}$$

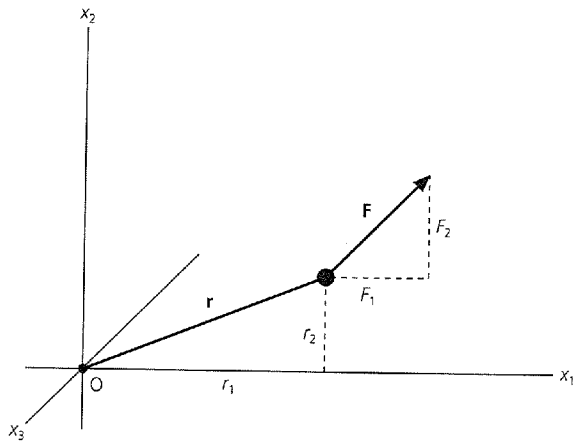


Fig. A.3-8 The x_3 component of the vector product $\boldsymbol{\tau} = \mathbf{r} \times \mathbf{F}$ gives the torque, $r_1 F_2 - r_2 F_1$ about the x_3 axis. In this case $r_1 F_2$ is greater than $r_2 F_1$, so counterclockwise rotation about the x_3 axis occurs.

For example, the torque about the x_3 axis is $\tau_3 = (r_1 F_2 - r_2 F_1)$, so each component of the force contributes a counterclockwise torque equal to the component times its lever arm, the perpendicular distance of the point from that axis (Fig. A.3-8).

Some useful identities, whose proofs are left as problems, are

$$\begin{aligned} \mathbf{a} \cdot (\mathbf{b} + \mathbf{c}) &= \mathbf{a} \cdot \mathbf{b} + \mathbf{a} \cdot \mathbf{c} \\ \mathbf{a} \times (\mathbf{b} + \mathbf{c}) &= \mathbf{a} \times \mathbf{b} + \mathbf{a} \times \mathbf{c} \\ \mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}) &= \mathbf{b} \cdot (\mathbf{c} \times \mathbf{a}) = \mathbf{c} \cdot (\mathbf{a} \times \mathbf{b}) \\ \mathbf{a} \times (\mathbf{b} \times \mathbf{c}) &= \mathbf{b}(\mathbf{a} \cdot \mathbf{c}) - \mathbf{c}(\mathbf{a} \cdot \mathbf{b}). \end{aligned} \tag{33}$$

A.3.5 Index notation

Vector equations, such as the definition of the cross product, can be cumbersome when written in terms of the components. Simplification can be obtained using *index notation*, whereby

an index assuming all possible values replaces the subscripts indicating coordinate axes. For example, the vector $\mathbf{u} = (u_1, u_2, u_3)$ is written u_i, where i can be 1, 2, or 3. In this notation, the scalar product is

$$\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2 + a_3 b_3 = \sum_{i=1}^{3} a_i b_i . \tag{34}$$

Because the sum over all coordinates appears frequently, the *Einstein summation convention* is often used, whereby an index repeated twice implies a summation over that index, and the summation sign is not explicitly written. Hence the scalar product of two real vectors is written

$$\mathbf{a} \cdot \mathbf{b} = a_i b_i, \tag{35}$$

using implied summation over the repeated index i. Similarly, the square of the magnitude of a real vector is

$$|\mathbf{u}|^2 = u_i u_i. \tag{36}$$

A repeated index is called a "dummy" index, like a dummy variable of integration, because it is used only within the summation. The form of the expression indicates that $u_i u_i$ is a scalar; because the repeated index is summed, no index remains "free." By contrast, u_i is a vector, because there is a free index.

Index notation is further simplified by introducing two symbols, δ_{ij} and ε_{ijk}. The *Kronecker delta*, δ_{ij}, is defined

$$\begin{aligned} \delta_{ij} &= 0 \quad \text{if } i \neq j, \\ &= 1 \quad \text{if } i = j. \end{aligned} \tag{37}$$

So, for example, $\delta_{11} = 1$, but $\delta_{12} = 0$. Using the Kronecker delta symbol, the relations between the Cartesian basis vectors (Eqns 25, 26) can be written compactly as

$$\hat{\mathbf{e}}_i \cdot \hat{\mathbf{e}}_j = \delta_{ij}. \tag{38}$$

The Kronecker delta, a function of two discrete variables i and j, is analogous to the Dirac delta function which is a function of a continuous variable (Section 6.2.5).

The *permutation symbol*, ε_{ijk}, is defined as

$$\begin{aligned} \varepsilon_{ijk} &= 0 \quad \text{if any of the indices are the same,} \\ &= 1 \quad \text{if } i, j, k \text{ are in order, i.e., } (1, 2, 3), (2, 3, 1), \\ &\qquad \text{or } (3, 1, 2) \\ &= -1 \quad \text{if } i, j, k \text{ are out of order,} \\ &\qquad \text{i.e., } (2, 1, 3), (3, 2, 1), (1, 3, 2). \end{aligned} \tag{39}$$

Cases where the indices are in order are known as even, or cyclic, permutations of the indices; those in which the indices are out of order are known as odd permutations. Because of the symmetries in the definition, $\varepsilon_{ijk} = \varepsilon_{jki} = \varepsilon_{kij}$. A useful relation, whose proof is left for the problems, is

$$\varepsilon_{ijk}\varepsilon_{ist} = \delta_{js}\delta_{kt} - \delta_{jt}\delta_{ks}. \tag{40}$$

Using index notation, the definition of the vector product (Eqn 28) becomes

$$(\mathbf{a} \times \mathbf{b})_i = \sum_{j=1}^{3} \sum_{k=1}^{3} \varepsilon_{ijk} a_j b_k = \varepsilon_{ijk} a_j b_k, \tag{41}$$

where the last form uses the summation convention. The notation shows that the cross product yields a vector because only one index, i, remains free after the repeated indices j and k are summed. To see that the index notation gives the correct definition, we expand the $i = 2$ component as

$$\begin{aligned} (\mathbf{a} \times \mathbf{b})_2 &= \varepsilon_{211} a_1 b_1 + \varepsilon_{212} a_1 b_2 + \varepsilon_{213} a_1 b_3 + \varepsilon_{221} a_2 b_1 + \varepsilon_{222} a_2 b_2 \\ &\quad + \varepsilon_{223} a_2 b_3 + \varepsilon_{231} a_3 b_1 + \varepsilon_{232} a_3 b_2 + \varepsilon_{233} a_3 b_3 \\ &= (a_3 b_1 - a_1 b_3), \end{aligned} \tag{42}$$

because the only nonzero ε_{ijk} terms are $\varepsilon_{213} = -1$ and $\varepsilon_{231} = 1$.

Index notation points out an interesting feature of the vector product. Because $a_i b_i = b_i a_i$, the scalar product commutes. By contrast, the properties of the permutation symbol show that

$$\mathbf{a} \times \mathbf{b} = \varepsilon_{ijk} a_j b_k = -\varepsilon_{ijk} b_j a_k = -\mathbf{b} \times \mathbf{a}, \tag{43}$$

so the order matters for the vector product.

Although index notation seems unnatural at first, it does more than simply shorten expressions. The notation explicitly indicates what operations must be performed, and thus makes them easier to evaluate. For example, suppose we seek to show that the cross product of a vector with itself is zero. In contrast to $(\mathbf{a} \times \mathbf{a})$, the notation $\varepsilon_{ijk} a_j a_k$ shows how the cross product should be evaluated. Because $a_j a_k$ is symmetric in the indices j and k, the permutation symbol makes the terms involving any pair of j and k sum to zero. We will see that index notation makes the complicated expressions that we encounter in studying stress and strain easier to evaluate.

A.3.6 Vector spaces

These concepts for vectors can be generalized in several ways. In three dimensions any vector is a weighted combination of three basis vectors. The usual choice of basis vectors along coordinate axes is for simplicity. We could choose any three mutually orthogonal vectors, which need not be of unit length, to be the basis vectors. To see this, remember that a physical vector does not depend on the coordinate system.

Moreover, the idea of vectors in two- or three-dimensional space can be generalized to spaces with a larger number of dimensions. For example, given unit vectors

$$\begin{aligned} &\hat{\mathbf{e}}_1 = (1, 0, 0, 0, 0), \quad \hat{\mathbf{e}}_2 = (0, 1, 0, 0, 0), \quad \hat{\mathbf{e}}_3 = (0, 0, 1, 0, 0), \\ &\hat{\mathbf{e}}_4 = (0, 0, 0, 1, 0), \quad \hat{\mathbf{e}}_5 = (0, 0, 0, 0, 1), \end{aligned} \tag{44}$$

a vector $\mathbf{u}$ can be formed from the basis vectors and components

$$\mathbf{u} = u_1\hat{\mathbf{e}}_1 + u_2\hat{\mathbf{e}}_2 + u_3\hat{\mathbf{e}}_3 + u_4\hat{\mathbf{e}}_4 + u_5\hat{\mathbf{e}}_5 = (u_1, u_2, u_3, u_4, u_5). \quad (45)$$

This vector is defined in a five-dimensional space, with five axes each orthogonal to the others, because their scalar products are zero. Although this is difficult to visualize (or draw), the mathematics carries through directly from the three-dimensional case. N mutually orthogonal vectors thus provide a basis for an N-dimensional space.

These ideas are formalized in terms of vectors in a general *linear vector space*. For our purposes, a vector space is a collection of vectors $\mathbf{x}, \mathbf{y}, \mathbf{z}$, satisfying several criteria:

- The sum of any two vectors in the space is also in the space.
- Vector addition commutes: $\mathbf{x}+\mathbf{y}=\mathbf{y}+\mathbf{x}$.
- Vector addition is associative: $(\mathbf{x}+\mathbf{y})+\mathbf{z}=\mathbf{x}+(\mathbf{y}+\mathbf{z})$.
- There exists a unique vector $\mathbf{0}$ such that for all $\mathbf{x}$, $\mathbf{x}=\mathbf{x}+\mathbf{0}$.
- There exists a unique vector $-\mathbf{x}$ such that for all $\mathbf{x}$, $\mathbf{x}+(-\mathbf{x}) = 0$.
- Scalar multiplication is associative: $\alpha(\beta\mathbf{x}) = (\alpha\beta)\mathbf{x}$.
- Scalar multiplication is distributive: $\alpha(\mathbf{x}+\mathbf{y}) = \alpha\mathbf{x} + \alpha\mathbf{y}$ and $(\alpha+\beta)\mathbf{x} = (\alpha\mathbf{x}+\beta\mathbf{x})$.

A point worth considering is the number of independent vectors in a vector space. Given N vectors $\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^N$ in a linear vector space, a weighted sum $\sum \alpha_i \mathbf{x}^i$ is called a *linear combination*. The N vectors are *linearly independent* if

$$\sum_{i=1}^{N} \alpha_i \mathbf{x}^i = 0 \text{ only when all } \alpha_i = 0, \quad (46)$$

so that no vector can be expressed as a combination of the others. Otherwise, the vectors are *linearly dependent*, and one can be expressed as a linear combination of the others.

This idea corresponds to that of basis vectors. If N basis vectors are mutually orthogonal, they are linearly independent. Because any vector in an N-dimensional space is a linear combination of N linearly independent basis vectors, the basis vectors *span* the space. Thus the dimension of a vector space is the number of linearly independent vectors within it. For example, we cannot find four linearly independent vectors in three dimensions.

Though vector spaces sound abstract, they are useful in seismology. For example, in Chapter 2 we represent travelling waves by normal modes, which are orthogonal basis vectors in a vector space, so any wave is a weighted sum of them. The modes of a string (Section 2.2.5) form a Fourier series (Chapter 6), in which a function is expanded into sine and cosine functions that are the basis vectors of a vector space. A similar approach is also used for the modes of the spherical earth (Section 2.9). Vector space ideas are also used in inverting seismological observations to study earth structure (Chapter 7).

A.4 Matrix algebra

A.4.1 Definitions

Matrix algebra is a powerful tool often used to study systems of equations. As a result, it appears in seismological applications, including stresses and strains, locating earthquakes, and seismic tomography. We thus review some basic ideas, often stating results without proof and leaving proofs for the problems. Further discussion of these topics can be found in linear algebra texts.

Given a matrix A with m rows and n columns, called an $m \times n$ matrix,

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} \quad (1)$$

and a second matrix B, also with m rows and n columns, matrix addition is defined by

$$A + B = \begin{pmatrix} a_{11}+b_{11} & a_{12}+b_{12} & \cdots & a_{1n}+b_{1n} \\ a_{21}+b_{21} & a_{22}+b_{22} & \cdots & a_{2n}+b_{2n} \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ a_{m1}+b_{m1} & a_{m2}+b_{m2} & \cdots & a_{mn}+b_{mn} \end{pmatrix}. \quad (2)$$

The usual convention is to indicate matrices with capital letters and their elements with lower-case ones.

Matrix multiplication is defined such that for a matrix A that is $m \times n$ and a matrix B that is $n \times r$, the ij^{th} element of the $m \times r$ product matrix $C = AB$ is defined by

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj} = a_{ik} b_{kj}. \quad (3)$$

The ij^{th} element of C is the scalar product of the ith row of A and the jth column of B. As a result, for matrix multiplication the two matrices need not have the same number of rows and columns, but must have the number of columns in the first matrix equal to the number of rows in the second. Often the numbers of rows and columns in the two matrices allow multiplication in only one order. Thus, in the example above, A "premultiplies" B, or B "postmultiplies" A. A convenient way to remember this is that the number of columns in the first matrix must equal the number of rows in the second, but this dimension does not appear in the product. In the case of $AB = C$, written schematically, we have $[m \times n][n \times r] = [m \times r]$. Hence, in the final form in Eqn 3, the summation convention shows that k is summed out, leaving i and j as free indices, so c_{ij} is a matrix element. Furthermore, even if both AB and BA are allowed, the two products are generally not equal, so matrix multiplication is not commutative.

The *identity matrix*, I, is a square matrix (one with the same number of rows and columns) whose diagonal elements are equal to 1 while all other elements are 0:

$$I = \begin{pmatrix} 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ . & . & \dots & . & . \\ . & . & \dots & . & . \\ 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & \dots & 0 & 1 \end{pmatrix}. \tag{4}$$

The identity matrix has the property that for any square matrix A,

$$AI = IA = A. \tag{5}$$

The *transpose* of a matrix A, A^T, is derived by placing the rows of A into the columns of A^T, so for $C = A^T$,

$$c_{ij} = a_{ji}. \tag{6}$$

The transpose has the properties that for matrices A and B,

$$(A+B)^T = A^T + B^T \quad \text{and} \quad (AB)^T = B^T A^T. \tag{7}$$

With these definitions, vector operations can be expressed using matrix algebra, by treating vectors as matrices with one column. For example, premultiplication of a vector by a matrix yields another vector, $\mathbf{y} = A\mathbf{x}$, such that

$$y_i = \sum_j a_{ij} x_j \quad \text{or} \quad y_i = a_{ij} x_j, \tag{8}$$

where the second form uses the summation convention. Each component y_i is the scalar product of the ith row of A with $\mathbf{x}$. Similarly, the scalar product of two vectors is given by the matrix product

$$\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^T \mathbf{b} = \sum_i a_i b_i = a_i b_i. \tag{9}$$

Thus the scalar product of two vectors yields a scalar, because a $1 \times m$ matrix times an $m \times 1$ matrix is a 1×1 matrix, or single value. The squared magnitude of a real vector can be written as

$$|\mathbf{u}|^2 = \mathbf{u} \cdot \mathbf{u} = \mathbf{u}^T \mathbf{u} = \sum_i u_i u_i = u_i u_i. \tag{10}$$

For vectors with complex components, the scalar product (Eqn A.3.23) is

$$\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^{*T} \mathbf{b} = \sum_i a_i^* b_i = a_i^* b_i. \tag{11}$$

This brings us to a minor point of notation. In linear algebra, as in the last few equations, it is common to treat vectors as column vectors represented by $n \times 1$ matrices with n rows and one column

$$\mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \\ . \\ . \\ u_n \end{pmatrix}, \tag{12}$$

whose transposes are row vectors (one row, n columns) like

$$\mathbf{u}^T = (u_1, u_2, \dots u_n). \tag{13}$$

Nonetheless, to save space, we sometimes write

$$\mathbf{u} = (u_1, u_2, \dots u_n), \tag{14}$$

while treating $\mathbf{u}$ as a column vector when required. Strictly speaking, we should call the row vector $\mathbf{u}^T$.

We often encounter matrices that are *symmetric*, or equal their transposes,

$$A = A^T, \quad a_{ij} = a_{ji}. \tag{15}$$

For a matrix A with complex elements, the conjugate matrix A^* is formed by taking the conjugate of each element, and the transpose is generalized to the *adjoint* matrix $A^+ = A^{*T}$, which is the complex conjugate of A^T. Note that if the elements of A are real, $A^+ = A^T$. A matrix A is *Hermitian* if it equals its adjoint

$$A = A^+, \quad a_{ij} = a_{ji}^*. \tag{16}$$

If A is real, "Hermitian" and "symmetric" are equivalent.

A.4.2 Determinant

A useful entity is the *determinant* of a matrix, written det A, or $|A|$. For an $n \times n$ matrix,

$$\det A = \sum_{j_1=1}^{n} \sum_{j_2=1}^{n} \dots \sum_{j_n=1}^{n} s(j_1, j_2, \dots j_n) a_{1j_1} a_{2j_2} \dots a_{nj_n}. \tag{17}$$

This complicated sum over n indices, $j_1, j_2, \dots j_n$, uses a generalized form of the permutation symbol

$$s(j_1, j_2, \dots j_n) = \text{sgn} \prod_{1 \le p < q \le n} (j_q - j_p). \tag{18}$$

The sgn function is one times the sign of its argument, so that it equals 1 if its argument is positive, -1 if its argument is negative, and 0 if its argument is zero. For $n = 3$,

$$s(j_1, j_2, j_3) = \text{sgn}\,[(j_2 - j_1)(j_3 - j_1)(j_3 - j_2)], \tag{19}$$

so that, for example,

$$s(1,2,3)=1, \quad s(2,1,3)=-1, \quad s(1,1,3)=0. \tag{20}$$

Because $s(j_1, j_2, j_3)$ suppresses terms with two equal indices, and assigns others a sign depending on the order of the indices, it is the same as the permutation symbol, $\varepsilon_{j_1 j_2 j_3}$ (Eqn A.3.39).

The definition of the determinant gives the familiar result for $n=2$:

$$\begin{aligned} |A| &= \det\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} = \sum_{j_1=1}^{2}\sum_{j_2=1}^{2} s(j_1,j_2)a_{1j_1}a_{2j_2} \\ &= s(1,1)a_{11}a_{21} + s(1,2)a_{11}a_{22} + s(2,1)a_{12}a_{21} + s(2,2)a_{12}a_{22} \\ &= a_{11}a_{22} - a_{12}a_{21}, \end{aligned} \tag{21}$$

because $s(1, 1) = s(2, 2) = 0$, $s(1, 2) = 1$, and $s(2, 1) = -1$. For a matrix with only one element, the determinant equals the matrix element.

Among the properties of determinants that we will find useful in solving systems of equations are:

- The determinant of a matrix equals that of its transpose, $|A| = |A^T|$.
- If two rows or columns of a matrix are interchanged, the determinant has the same absolute value but changes sign.
- If one row (or column) is multiplied by a constant, the determinant is multiplied by that constant.
- If a multiple of one row (or column) is added to another row (or column), the determinant is unchanged.
- If two rows or columns of a matrix are the same, the determinant is zero.

Proving these properties is left for the problems.

A.4.3 Inverse

For an $n \times n$ square matrix A, the *inverse* matrix A^{-1} is defined such that multiplication by the inverse gives the identity matrix

$$A^{-1}A = AA^{-1} = I. \tag{22}$$

A^{-1} can be written in terms of the *cofactor matrix*, C, whose elements

$$c_{ij} = (-1)^{i+j}|A_{ij}| \tag{23}$$

are formed from the determinants of A_{ij}, an $(n-1)\times(n-1)$ square matrix formed by deleting the ith row and j^{th} column from A. If $|A|$ is not zero,

$$A^{-1} = C^T/|A|. \tag{24}$$

For the familiar $n=2$ case, see problem 7.

A matrix whose determinant is zero does not have an inverse, and is called *singular*. Because the determinant of a matrix with two equal rows or columns is zero, such a matrix is singular. More generally, a matrix is singular if a row or column is a linear combination of the others.

The inverse of the matrix product AB, if AB is nonsingular, obeys

$$(AB)^{-1} = B^{-1}A^{-1}. \tag{25}$$

A matrix A whose transpose equals its inverse,

$$A^{-1} = A^T, \tag{26}$$

is called *orthogonal*. By extension, a matrix A with complex elements is *unitary* if its adjoint and inverse are equal

$$A^{-1} = A^+. \tag{27}$$

A.4.4 Systems of linear equations

A vector–matrix representation is often used for systems of linear equations. In this formulation, a system of m equations for n unknown variables x_i,

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 \ldots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 \ldots + a_{2n}x_n &= b_2 \\ &\ldots \\ a_{m1}x_1 + a_{m2}x_2 \ldots + a_{mn}x_n &= b_m \end{aligned} \tag{28}$$

is written in the form

$$\sum_{j=1}^{n} a_{ij}x_j = b_i \quad \text{or} \quad A\mathbf{x} = \mathbf{b}, \tag{29}$$

by defining the matrix of coefficients and column vectors for the unknowns and right-hand side,

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} \quad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \cdot \\ \cdot \\ \cdot \\ b_m \end{pmatrix}. \tag{30}$$

The coefficient matrix A is $m \times n$, because there is one row for each equation, and one column for each unknown.

The $A\mathbf{x} = \mathbf{b}$ form illustrates that whether a system of equations can be solved depends on the matrix A. A system of equations is called *homogeneous* in the special case that $\mathbf{b} = 0$, and *inhomogeneous* for all other cases in which $\mathbf{b} \neq 0$. We consider here only systems where the number of unknowns and equations are equal, so the coefficient matrix A is square. If A possesses an inverse, both sides can be premultiplied by A^{-1}, and

$$A^{-1}A\mathbf{x} = A^{-1}\mathbf{b} = I\mathbf{x} = \mathbf{x} \tag{31}$$

yields a unique solution vector **x**. For inhomogeneous systems, computing A^{-1} provides a straightforward manner of solving for the unknown variables x_i. For homogeneous systems of equations, the equation shows that $\mathbf{x} = 0$ if A^{-1} exists. Thus, for a homogeneous system to have a nonzero or *nontrivial* solution, A must be singular. This occurs if the determinant of A is zero, implying that some of the rows (or columns) of A are not linearly independent. If a nontrivial solution of the homogeneous system exists, any constant times that solution is also a solution.

If the coefficient matrix is singular, the corresponding inhomogeneous system of equations does not have unique solutions, and may have none. The existence of A^{-1} and the solvability of the equations thus depend on whether the rows and columns of A are linearly independent. For example, if the rows are linearly dependent, there are fewer independent equations than unknowns and difficulties result, as discussed in the context of inverse problems (Chapter 7).

A.4.5 *Solving systems of equations on a computer*

Standard methods exist to solve linear equations on a computer. Consider the basic problem

$$A\mathbf{x} = \mathbf{b} \tag{32}$$

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

in which we solve for **x**, given A and **b**. If A were a *triangular* matrix T, with zeroes below the diagonal, it would be easy to solve the system

$$T\mathbf{x} = \mathbf{d} \tag{33}$$

$$\begin{pmatrix} t_{11} & t_{12} & t_{13} \\ 0 & t_{22} & t_{23} \\ 0 & 0 & t_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix}$$

by starting with the simplest (bottom) equation, solving for x_3, and solving the other equations in succession to find x_2 and then x_1. In other words, the solution

$$x_3 = d_3/t_{33} \tag{34}$$

can be substituted into the middle equation to find

$$x_2 = (d_2 - t_{23}x_3)/t_{22}. \tag{35}$$

Then, by substituting x_3 and x_2 into the first equation,

$$x_1 = (d_1 - t_{13}x_3 - t_{12}x_2)/t_{11}. \tag{36}$$

The importance of this idea is that an arbitrary matrix can be triangularized. Consider that the solution of the system of equations is not changed by any of the following *elementary row operations*:

(i) Rearranging the equations, which corresponds to interchanging rows in the **b** vector and matrix, i.e.,

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{31} & a_{32} & a_{33} \\ a_{21} & a_{22} & a_{23} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_3 \\ b_2 \end{pmatrix}. \tag{37}$$

The solution is unchanged because the order of the equations is arbitrary.

(ii) Multiplying an equation by a constant c, which corresponds to multiplying a row of A and the corresponding element of **b** by a constant, i.e.,

$$\begin{pmatrix} ca_{11} & ca_{12} & ca_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} cb_1 \\ b_2 \\ b_3 \end{pmatrix}. \tag{38}$$

(iii) Adding two equations, which corresponds to adding a multiple of one row to another, i.e.,

$$\begin{pmatrix} ca_{11} + a_{21} & ca_{12} + a_{22} & ca_{13} + a_{23} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} cb_1 + b_2 \\ b_2 \\ b_3 \end{pmatrix}. \tag{39}$$

Thus if the system $A\mathbf{x} = \mathbf{b}$ is transformed into $T\mathbf{x} = \mathbf{d}$ using elementary row operations, the two systems of equations have the same solutions **x**. This provides a fast method of solving the system: combine A and **b** into a single *augmented matrix*

$$(A, \mathbf{b}) = \begin{pmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ a_{31} & a_{32} & a_{33} & b_3 \end{pmatrix} \tag{40}$$

and triangularize the augmented matrix to obtain

$$(T, \mathbf{d}) = \begin{pmatrix} t_{11} & t_{12} & t_{13} & d_1 \\ 0 & t_{22} & t_{23} & d_2 \\ 0 & 0 & t_{33} & d_3 \end{pmatrix}, \tag{41}$$

which represents a set of equations easily solved for **x** by the method in Eqns 34–6.

The matrix is triangularized using the following method column by column:

- Find the element of maximum absolute value in the column on or below the diagonal.
- If this "pivot" element is below the diagonal, interchange rows to get it on the diagonal.

- Subtract multiples of the pivot row from rows below it to get zeroes below the diagonal.

The pivoting, though not absolutely necessary, avoids possible numerical difficulties. Note that once a column is zeroed below the diagonal, we do not have to think about it any more.

For an illustration of this method, called *Gaussian elimination with partial pivoting*, consider solving the system of equations

$$\begin{aligned} x_1 + x_2 &= 5, \\ 4x_1 + x_2 + x_3 &= 4, \\ 2x_1 + 2x_2 + 2x_3 &= 3. \end{aligned} \tag{42}$$

This can be expressed in matrix form as

$$\begin{pmatrix} 1 & 1 & 0 \\ 4 & 1 & 1 \\ 2 & 2 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 5 \\ 4 \\ 3 \end{pmatrix}, \tag{43}$$

and solved by triangularizing the augmented matrix

$$\begin{pmatrix} 1 & 1 & 0 & 5 \\ 4 & 1 & 1 & 4 \\ 2 & 2 & 2 & 3 \end{pmatrix}. \tag{44}$$

To get zeroes below the diagonal in the first column, we first move 4, the element with the largest absolute value in the first column, to the diagonal by interchanging rows

$$\begin{pmatrix} 4 & 1 & 1 & 4 \\ 1 & 1 & 0 & 5 \\ 2 & 2 & 2 & 3 \end{pmatrix}. \tag{45}$$

We then subtract 1/4 times the first row from second, and 1/2 times the first row from third, leaving

$$\begin{pmatrix} 4 & 1 & 1 & 4 \\ 0 & 0.75 & -0.25 & 4 \\ 0 & 1.5 & 1.5 & 1 \end{pmatrix}. \tag{46}$$

Next, to zero the elements below the diagonal in the second column, we interchange rows to get the pivot for this column, 1.5, on the diagonal:

$$\begin{pmatrix} 4 & 1 & 1 & 4 \\ 0 & 1.5 & 1.5 & 1 \\ 0 & 0.75 & -0.25 & 4 \end{pmatrix} \tag{47}$$

and subtract $0.75/1.5 = 0.5$ times the second row from the third

$$\begin{pmatrix} 4 & 1 & 1 & 4 \\ 0 & 1.5 & 1.5 & 1 \\ 0 & 0 & -1 & 3.5 \end{pmatrix} \tag{48}$$

to complete the triangularization. We then solve the equations for $\mathbf{x}$, beginning with the bottom one, as in Eqns 34–6.

A similar procedure can be used to invert a matrix. This method uses the idea that two vector–matrix equations

$$A\mathbf{x} = \mathbf{b} \quad \text{and} \quad A\mathbf{y} = \mathbf{c} \tag{49}$$

can be combined into one by forming an augmented matrix from each pair of vectors,

$$X = (\mathbf{x}, \mathbf{y}), \quad B = (\mathbf{b}, \mathbf{c}), \tag{50}$$

and writing the matrix equation

$$AX = B. \tag{51}$$

Because $\mathbf{x}$, the solution to $A\mathbf{x} = \mathbf{b}$, is not changed by elementary row operations on the augmented matrix $(A, \mathbf{b})$, the corresponding solution to $AX = B$ is unaffected by elementary row operations on the augmented matrix (A, B).

To apply this to matrix inversion, consider a special case

$$AX = I, \tag{52}$$

whose solution $X = A^{-1}$ is the inverse of the $n \times n$ matrix A. X is unaffected by elementary row operations that convert the augmented matrix

$$(A, I) = \begin{pmatrix} a_{11} & . & . & a_{1n} & 1 & . & . & 0 \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ a_{n1} & . & . & a_{nn} & 0 & . & . & 1 \end{pmatrix} \tag{53}$$

to one whose left side is the identity

$$(I, B) = \begin{pmatrix} 1 & . & . & 0 & b_{11} & . & . & b_{1n} \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ 0 & . & . & 1 & b_{n1} & . & . & b_{nn} \end{pmatrix}, \tag{54}$$

so the corresponding equation

$$IX = B \tag{55}$$

shows that the right side of the matrix gives $B = X = A^{-1}$, the inverse of A. The sequence of operations used to diagonalize the left (A) side of the augmented matrix (A, I) are similar to those that triangularize a matrix.

A.5 Vector transformations

In seismology, we often apply two types of transformations to vectors. In the first, the same vector is expressed in two

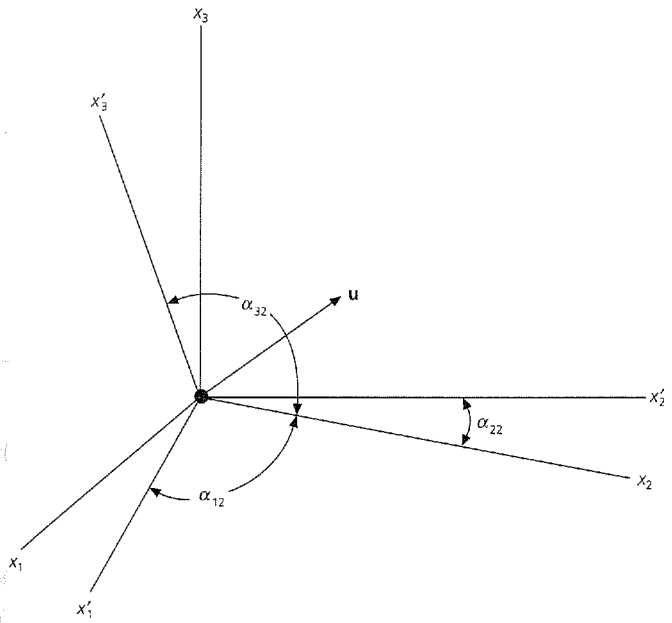


Fig. A.5-1 The relation between two orthogonal coordinate systems with the same origin is described by the angles α_{ij} between the two sets of axes.

different coordinate systems. In the second, some operation converts a vector to another vector expressed in the same coordinate system. In this section we summarize these transformations and their differences.

A.5.1 *Coordinate transformations*

We have seen that vectors remain the same regardless of the coordinate system in which they are defined, although their components differ between coordinate systems. Thus vectors can be defined in one coordinate system (for example, one oriented along an earthquake fault plane) and reexpressed in another (such as a geographic coordinate system). This property is very useful for solving problems and gives valuable insight into the nature of vectors.

To define the relation between vector components and coordinate systems, consider two orthogonal Cartesian coordinate systems (Fig. A.5-1). Because the origins are the same, one coordinate system can be obtained by rotating the other through three angles. The relation between the two sets of unit basis vectors, $\hat{e}_1, \hat{e}_2, \hat{e}_3$ and $\hat{e}_1', \hat{e}_2', \hat{e}_3'$, is given by their scalar products, called *direction cosines*,

$$\hat{e}_i' \cdot \hat{e}_j = \cos \alpha_{ij} = a_{ij}, \tag{1}$$

where the angles α_{ij} are the angles between the two sets of axes.

A vector can be expressed in terms of its components in the two coordinate systems

$$\mathbf{u} = u_1\hat{e}_1 + u_2\hat{e}_2 + u_3\hat{e}_3 = u_1'\hat{e}_1' + u_2'\hat{e}_2' + u_3'\hat{e}_3'. \tag{2}$$

Given the components u_i in the unprimed system, the components u_i' in the primed system are found by taking the scalar products of the vector with the basis vectors of the primed system:

$$\begin{aligned} u_1' &= \hat{e}_1' \cdot \mathbf{u} = (\hat{e}_1' \cdot \hat{e}_1)u_1 + (\hat{e}_1' \cdot \hat{e}_2)u_2 + (\hat{e}_1' \cdot \hat{e}_3)u_3 \\ &= a_{11}u_1 + a_{12}u_2 + a_{13}u_3, \\ u_2' &= \hat{e}_2' \cdot \mathbf{u} = a_{21}u_1 + a_{22}u_2 + a_{23}u_3, \\ u_3' &= \hat{e}_3' \cdot \mathbf{u} = a_{31}u_1 + a_{32}u_2 + a_{33}u_3. \end{aligned} \tag{3}$$

These can be written as a matrix equation

$$\mathbf{u}' = A\mathbf{u}, \quad \text{or} \quad \begin{pmatrix} u_1' \\ u_2' \\ u_3' \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}, \tag{4}$$

where A is the matrix that transforms a vector from the unprimed to the primed system. Note that this is not a relation between two different vectors $\mathbf{u}$ and $\mathbf{u}'$ — it is a relationship between the *components* of the *same* vector in two coordinate systems. It turns out that the matrix A uniquely describes the transformation between these coordinate systems.

For example, a unit basis vector for the unprimed system

$$\hat{e}_1 = 1\hat{e}_1 + 0\hat{e}_2 + 0\hat{e}_3 = (1, 0, 0) \tag{5}$$

has components in the primed system given by

$$\begin{pmatrix} a_{11} \\ a_{21} \\ a_{31} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \tag{6}$$

and so is written

$$a_{11}\hat{e}_1' + a_{21}\hat{e}_2' + a_{31}\hat{e}_3' = (a_{11}, a_{21}, a_{31}) \tag{7}$$

in the primed system. The last expression is just the first column of A. Similarly, the components of $\hat{e}_2$ and $\hat{e}_3$ in the primed system are the second and third columns of A, respectively. Thus the columns of the transformation matrix A are the basis vectors of the unprimed system written in terms of their components in the primed system.

For example, consider rotating a Cartesian coordinate system by θ counterclockwise about the $\hat{e}_3$ axis, so that the only rotation occurs in the $\hat{e}_1$–$\hat{e}_2$ plane. The $\hat{e}_3$ axis is also the $\hat{e}_3'$ axis (Fig. A.5-2). The elements of the transformation matrix are found by evaluating the scalar products of the basis vectors $a_{ij} = \hat{e}_i' \cdot \hat{e}_j$, so

$$\begin{aligned} a_{11} &= \hat{e}_1' \cdot \hat{e}_1 = \cos\theta, & a_{12} &= \hat{e}_1' \cdot \hat{e}_2 = \cos(90° - \theta) = \sin\theta, \\ a_{22} &= \hat{e}_2' \cdot \hat{e}_2 = \cos\theta, & a_{21} &= \hat{e}_2' \cdot \hat{e}_1 = \cos(90° + \theta) = -\sin\theta, \\ a_{33} &= \hat{e}_3' \cdot \hat{e}_3 = 1, & a_{13} &= a_{23} = a_{31} = a_{32} = 0, \end{aligned} \tag{8}$$

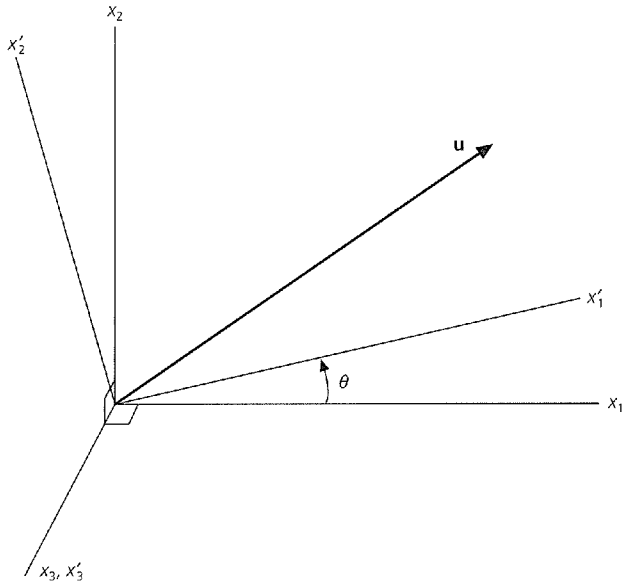


Fig. A.5-2 The relation between the axes of two orthogonal coordinate systems differing by a rotation θ in the x_1–x_2 plane.

and the components of a vector in the two systems are related by

$$\begin{pmatrix} u_1' \\ u_2' \\ u_3' \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}. \tag{9}$$

Thus the $\hat{e}_1$ and $\hat{e}_1'$, and the $\hat{e}_2$ and $\hat{e}_2'$ components differ, whereas the $\hat{e}_3$ and $\hat{e}_3'$ components are the same. To check this, consider the case where $\theta = 90°$. As expected, (1, 0, 0) in the unprimed system becomes (0, −1, 0) in the primed system, and (0, 1, 0) in the unprimed system becomes (1, 0, 0) in the primed system, while (0, 0, 1) in the unprimed system remains (0, 0, 1) in the primed system.

Seismologists often use such a geometry. Because the ground motion is a vector, seismometers are generally oriented to record its components in the east–west, north–south, and up–down directions. This decomposition is less useful than decomposing ground motion into its *radial* and *transverse* components, those along and perpendicular to the great circle connecting the earthquake and seismometer. The vertical component is useful as is, so a rotation about the vertical by the angle between East and the great circle connecting the earthquake and seismometer converts the E–W and N–S components into the new representation. The relevant angle, the *back azimuth* to the source from the receiver, is discussed in Section A.7.2.

We can also reverse the transformation. By analogy to Eqn 3, the components in the unprimed system can be found from those in the primed system as

$$\begin{aligned} u_1 &= \hat{e}_1 \cdot \mathbf{u}' = (\hat{e}_1 \cdot \hat{e}_1')u_1' + (\hat{e}_1 \cdot \hat{e}_2')u_2' + (\hat{e}_1 \cdot \hat{e}_3')u_3' \\ &= a_{11}u_1' + a_{21}u_2' + a_{31}u_3', \\ u_2 &= \hat{e}_2 \cdot \mathbf{u}' = a_{12}u_1' + a_{22}u_2' + a_{32}u_3', \\ u_3 &= \hat{e}_3 \cdot \mathbf{u}' = a_{13}u_1' + a_{23}u_2' + a_{33}u_3'. \end{aligned} \tag{10}$$

Combining these to express the reverse transformation in vector–matrix form,

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \\ a_{13} & a_{23} & a_{33} \end{pmatrix} \begin{pmatrix} u_1' \\ u_2' \\ u_3' \end{pmatrix}, \tag{11}$$

shows that the reverse transformation matrix is just the transpose of the transformation matrix A

$$\mathbf{u} = A^T\mathbf{u}'. \tag{12}$$

Hence a unit basis vector in the primed system

$$\hat{e}_1' = 1\hat{e}_1' + 0\hat{e}_2' + 0\hat{e}_3' \tag{13}$$

becomes, by the matrix transformation,

$$a_{11}\hat{e}_1 + a_{12}\hat{e}_2 + a_{13}\hat{e}_3 \tag{14}$$

in the unprimed system. This is the first row of A, so the rows of A are the primed basis vectors expressed in the unprimed coordinates. This is natural because the transformations are related by the matrix transpose.

Alternatively, the reverse transformation can be found directly by starting with $\mathbf{u}' = A\mathbf{u}$ and multiplying both sides by the inverse matrix

$$A^{-1}\mathbf{u}' = A^{-1}A\mathbf{u} = \mathbf{I}\mathbf{u} = \mathbf{u}. \tag{15}$$

Comparison with Eqn 12 shows that the inverse of the transformation matrix equals its transpose, so the transformation matrix is an orthogonal matrix. This seems reasonable because the columns of A, which represent orthogonal basis vectors, are orthogonal. Similarly, the rows of A are orthogonal. As a result, such coordinate transformations are called *orthogonal transformations*. An important feature of orthogonal transformations, whose proof is left as a homework problem, is that they preserve the length of vectors.

The transformation relations, Eqns 4 and 12, provide a mathematical definition of a vector. Any vector must transform between coordinate systems in this way. A set of three entities defined at points in space (for example, temperature, pressure, and density) that does not obey the transformation equations is not a vector.

A.5.2 Eigenvalues and eigenvectors

The product of an arbitrary $n \times n$ matrix A and an arbitrary n-component vector $\mathbf{x}$

$$\mathbf{y} = A\mathbf{x} \tag{16}$$

is also a vector in n dimensions. This is not the same as coordinate transformation; the vector $\mathbf{x}$ is transformed into another distinct vector, with both vectors expressed in the same coordinate system.

A physically important class of transformations convert a vector into one parallel to the original vector, so that

$$A\mathbf{x} = \lambda\mathbf{x}, \tag{17}$$

where A is a matrix, and λ is a scalar. The only effect of the transformation is that the length of $\mathbf{x}$ changes by a factor of λ. For a given A, it is useful to know which vectors $\mathbf{x}$ and scalars λ satisfy this equation.

In three dimensions, the case most commonly encountered, Eqn 17 can be written

$$(A - \lambda I)\mathbf{x} = 0$$

$$\begin{pmatrix} a_{11}-\lambda & a_{12} & a_{13} \\ a_{21} & a_{22}-\lambda & a_{23} \\ a_{31} & a_{32} & a_{33}-\lambda \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \tag{18}$$

This is a homogeneous system of linear equations, so nontrivial solutions exist only if the matrix $(A - \lambda I)$ is singular. We thus seek values of λ such that the determinant

$$|(A - \lambda I)| = \det\begin{pmatrix} a_{11}-\lambda & a_{12} & a_{13} \\ a_{21} & a_{22}-\lambda & a_{23} \\ a_{31} & a_{32} & a_{33}-\lambda \end{pmatrix} = 0. \tag{19}$$

Evaluating the determinant gives the *characteristic polynomial*

$$\lambda^3 - I_1\lambda^2 + I_2\lambda - I_3 = 0, \tag{20}$$

which depends on three constants called the *invariants* of A:

$$I_1 = a_{11} + a_{22} + a_{33},$$

$$I_2 = \det\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} + \det\begin{pmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{pmatrix} + \det\begin{pmatrix} a_{11} & a_{13} \\ a_{31} & a_{33} \end{pmatrix},$$

$$I_3 = \det A. \tag{21}$$

I_1, the first invariant, or *trace*, of A, is the sum of the diagonal elements of A. The invariants of a matrix have significance for stresses, strains, and earthquake moment tensors, because they are not changed by orthogonal transformations.

The characteristic polynomial is a cubic equation in λ with three roots, or *eigenvalues*, λ_m for which the determinant $|A - \lambda I|$ is zero. For each eigenvalue there is an associated nontrivial *eigenvector*, $\mathbf{x}^{(m)}$, satisfying

$$A\mathbf{x}^{(m)} = \lambda_m\mathbf{x}^{(m)}. \tag{22}$$

The components of the eigenvector, $x_1^{(m)}$, $x_2^{(m)}$, $x_3^{(m)}$, are found by solving

$$\begin{pmatrix} a_{11}-\lambda_m & a_{12} & a_{13} \\ a_{21} & a_{22}-\lambda_m & a_{23} \\ a_{31} & a_{32} & a_{33}-\lambda_m \end{pmatrix}\begin{pmatrix} x_1^{(m)} \\ x_2^{(m)} \\ x_3^{(m)} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \tag{23}$$

Each eigenvalue and its associated eigenvector form a pair satisfying Eqn 22. In general, an eigenvalue and the eigenvector associated with a different eigenvalue will not satisfy the equation.

For example, the eigenvalues of

$$A = \begin{pmatrix} 3 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 3 \end{pmatrix} \tag{24}$$

are found by solving the characteristic polynomial

$$\lambda^3 - 8\lambda^2 + 19\lambda - 12 = 0, \tag{25}$$

whose roots are $\lambda_1 = 4$, $\lambda_2 = 3$, $\lambda_3 = 1$. Next, the equations

$$\begin{pmatrix} 3-\lambda_m & -1 & 0 \\ -1 & 2-\lambda_m & -1 \\ 0 & -1 & 3-\lambda_m \end{pmatrix}\begin{pmatrix} x_1^{(m)} \\ x_2^{(m)} \\ x_3^{(m)} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{26}$$

are solved for each eigenvalue to yield the associated eigenvector. Thus for $\lambda_3 = 1$,

$$\begin{aligned} 2x_1^{(3)} - x_2^{(3)} &= 0, \\ -x_1^{(3)} + x_2^{(3)} - x_3^{(3)} &= 0, \\ -x_2^{(3)} + 2x_3^{(3)} &= 0. \end{aligned} \tag{27}$$

All three unknowns cannot be found uniquely, because these are homogeneous equations. We thus set $x_1^{(3)}$ equal to 1 and find the other two unknowns, $x_2^{(3)} = 2$, $x_3^{(3)} = 1$. Similarly, the other eigenvectors are found by substituting λ_2 and λ_1 in Eqn 26, so

$$\mathbf{x}^{(3)} = (1, 2, 1), \quad \mathbf{x}^{(2)} = (1, 0, -1), \quad \mathbf{x}^{(1)} = (1, -1, 1). \tag{28}$$

Because the eigenvectors are solutions to a set of homogeneous equations, any multiple of an eigenvector is also an eigenvector. The eigenvectors thus determine a direction in space, but the magnitude of the vector is arbitrary. Often the eigenvectors are normalized to unit magnitude. The set we have found can be written as

$$\begin{aligned} &\mathbf{x}^{(1)} = (1/\sqrt{3}, -1/\sqrt{3}, 1/\sqrt{3}), \quad \mathbf{x}^{(2)} = (1/\sqrt{2}, 0, -1/\sqrt{2}), \\ &\mathbf{x}^{(3)} = (1/\sqrt{6}, 2/\sqrt{6}, 1/\sqrt{6}). \end{aligned} \tag{29}$$

Sometimes complications arise, as for the matrix

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{30}$$

with eigenvalues 1, 1, and 0. Using the method given above to find the eigenvector for $\lambda_3 = 0$ by setting $x_1^{(3)} = 1$ yields no solution. Setting $x_2^{(3)} = 1$, however, yields a correct solution for the eigenvector, (0, 1, 0). Because this has no $\hat{\mathbf{e}}_1$ component, we could not have set $x_1^{(3)} = 1$ and found the other components.

This example illustrates a complication that arises for a *degenerate*, or repeated, eigenvalue: e.g., $\lambda_1 = \lambda_2 = 1$. In this case, the eigenvalue corresponds not to an eigenvector but to an entire plane, and any vector contained within it is an eigenvector. Two eigenvectors spanning this plane can be found by finding the eigenvector of the nondegenerate eigenvalue, and then choosing two independent vectors orthogonal to it. Because the eigenvector for the nondegenerate eigenvalue is (0, 1, 0), two possible orthogonal eigenvectors for the degenerate eigenvalue are (1, 0, 0) and (0, 0, 1).

A.5.3 Symmetric matrix eigenvalues, eigenvectors, diagonalization, and decomposition

The eigenvalues and eigenvectors of a symmetric matrix have interesting properties. An $n \times n$ matrix H has a characteristic polynomial of degree n, each of whose n roots is an eigenvalue. Consider two eigenvalues and their associated eigenvectors

$$H\mathbf{x}^{(i)} = \lambda_i \mathbf{x}^{(i)}, \quad H\mathbf{x}^{(j)} = \lambda_j \mathbf{x}^{(j)}. \tag{31}$$

Multiplication of the first equation by $\mathbf{x}^{(j)T}$ (the transpose of $\mathbf{x}^{(j)}$) and the second equation by $\mathbf{x}^{(i)T}$ yields

$$\mathbf{x}^{(j)T} H\mathbf{x}^{(i)} = \lambda_i \mathbf{x}^{(j)T}\mathbf{x}^{(i)}, \quad \mathbf{x}^{(i)T} H\mathbf{x}^{(j)} = \lambda_j \mathbf{x}^{(i)T}\mathbf{x}^{(j)}. \tag{32}$$

Transposing both sides of the second part of Eqn 32 and subtracting it from the first gives

$$\mathbf{x}^{(j)T} H\mathbf{x}^{(i)} - \mathbf{x}^{(j)T} H^T\mathbf{x}^{(i)} = (\lambda_i - \lambda_j)\mathbf{x}^{(j)T}\mathbf{x}^{(i)}. \tag{33}$$

Because H is symmetric, it equals its transpose, $H = H^T$, so the left-hand side is zero

$$0 = (\lambda_i - \lambda_j)\mathbf{x}^{(j)T}\mathbf{x}^{(i)}. \tag{34}$$

Thus, if $i \neq j$ and the two eigenvalues are different, their associated eigenvectors must be orthogonal so that their scalar product $\mathbf{x}^{(j)T}\mathbf{x}^{(i)}$ is zero. Thus, for a symmetric matrix, eigenvectors associated with distinct eigenvalues are orthogonal.

This result lets us diagonalize a symmetric matrix. To illustrate this for a 3×3 case, consider a matrix U whose columns are the eigenvectors of the symmetric matrix H

$$U = \begin{pmatrix} x_1^{(1)} & x_1^{(2)} & x_1^{(3)} \\ x_2^{(1)} & x_2^{(2)} & x_2^{(3)} \\ x_3^{(1)} & x_3^{(2)} & x_3^{(3)} \end{pmatrix}. \tag{35}$$

If the eigenvalues of H are distinct, the eigenvectors of H, and hence the columns of the eigenvector matrix, are orthogonal, so U is an orthogonal matrix satisfying $U^{-1} = U^T$.

The entire set of eigenvalue–eigenvector pairs, each of which satisfy $H\mathbf{x}^{(i)} = \lambda_i \mathbf{x}^{(i)}$, can be written as the matrix equation

$$HU = U\Lambda, \tag{36}$$

where Λ is the diagonal matrix with eigenvalues on the diagonal

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix}. \tag{37}$$

Premultiplying both sides of Eqn 36 by the inverse of the eigenvector matrix yields

$$U^{-1}HU = U^T HU = \Lambda, \tag{38}$$

which shows how the eigenvector matrix can be used to diagonalize a symmetric matrix. This result can also be stated as

$$H = U\Lambda U^T, \tag{39}$$

which illustrates how a symmetric matrix can be decomposed into a diagonal eigenvalue matrix and the orthogonal eigenvector matrix. Similar results apply for complex Hermitian matrices.

We will see that if a matrix contains the components of vectors expressed in a coordinate system, the physical problem under discussion can be simplified by diagonalizing the matrix. This corresponds to rewriting the problem in its "natural" coordinate system, whose basis set is the eigenvectors, an idea used in discussing stresses in the earth (Section 2.3.4) and the seismic moment tensor (Section 4.4.5).

A.6 Vector calculus

A.6.1 Scalar and vector fields

Many phenomena in seismology depend on how physical quantities vary in space. Some, like density or temperature, are *scalar fields*, scalar valued functions of the position vector $\mathbf{x}$ denoted by expressions like $\phi(\mathbf{x})$ or $\phi(x_1, x_2, x_3)$. Similarly, a vector that varies in space is described by a *vector field*. For example, seismic waves are described by the variation in the displacement vector

$$\mathbf{u}(\mathbf{x}) = \mathbf{u}(x_1, x_2, x_3)$$
$$= u_1(x_1, x_2, x_3)\hat{\mathbf{e}}_1 + u_2(x_1, x_2, x_3)\hat{\mathbf{e}}_2 + u_3(x_1, x_2, x_3)\hat{\mathbf{e}}_3 \qquad (1)$$

as a function of position, and result in turn from forces derived from spatial derivatives of the stress tensor.

Spatial variations of scalar, vector, or tensor fields are described using the vector differential operator "del", ∇,

$$\nabla = \left(\hat{\mathbf{e}}_1 \frac{\partial}{\partial x_1}, \hat{\mathbf{e}}_2 \frac{\partial}{\partial x_2}, \hat{\mathbf{e}}_3 \frac{\partial}{\partial x_3}\right). \qquad (2)$$

This operator has the form of a vector, but has meaning only when applied to a scalar, vector, or tensor field. We first review uses of the ∇ operator in Cartesian coordinates, and in the next section discuss the more complicated forms for spherical coordinates.

A.6.2 Gradient

The simplest application of the ∇ operator is the *gradient*, a vector field formed from the spatial derivatives of a scalar field. If $\phi(\mathbf{x})$ is a scalar function of position, the gradient is defined by

$$\text{grad}\ \phi(\mathbf{x}) = \nabla\phi(\mathbf{x}) = \frac{\partial\phi(\mathbf{x})}{\partial x_1}\hat{\mathbf{e}}_1 + \frac{\partial\phi(\mathbf{x})}{\partial x_2}\hat{\mathbf{e}}_2 + \frac{\partial\phi(\mathbf{x})}{\partial x_3}\hat{\mathbf{e}}_3, \qquad (3)$$

where $\partial\phi(\mathbf{x})/\partial x_1$ is the partial derivative of $\phi(x_1, x_2, x_3)$ with respect to x_1, for x_2 and x_3 held constant. The gradient is a vector field whose components equal the partial derivative with respect to the corresponding coordinate.

Expressions like Eqns 1 and 3 can be written more compactly if the dependences on position are not written explicitly, i.e.,

$$\nabla\phi = \frac{\partial\phi}{\partial x_1}\hat{\mathbf{e}}_1 + \frac{\partial\phi}{\partial x_2}\hat{\mathbf{e}}_2 + \frac{\partial\phi}{\partial x_3}\hat{\mathbf{e}}_3. \qquad (4)$$

In this notation, it is implicit that ϕ, its derivatives, and hence the gradient, vary with position.

For example, the elevation $\phi(x_1, x_2)$ is a scalar field describing the topography as a function of position in a two-dimensional region. This is often plotted using topographic contours (Fig. A.6-1), curves along which ϕ is constant. At any point, $\partial\phi/\partial x_1$ is the slope in the x_1 direction, and $\partial\phi/\partial x_2$ is the slope in the x_2 direction.

The gradient can be used to find the slope in any direction. The projection of a vector in a given direction is the scalar product of the vector and the unit normal vector in that direction, $\hat{\mathbf{n}} = (n_1, n_2)$. Thus the scalar product of the gradient with the normal vector,

$$\hat{\mathbf{n}} \cdot \nabla\phi = n_1\frac{\partial\phi}{\partial x_1} + n_2\frac{\partial\phi}{\partial x_2}, \qquad (5)$$

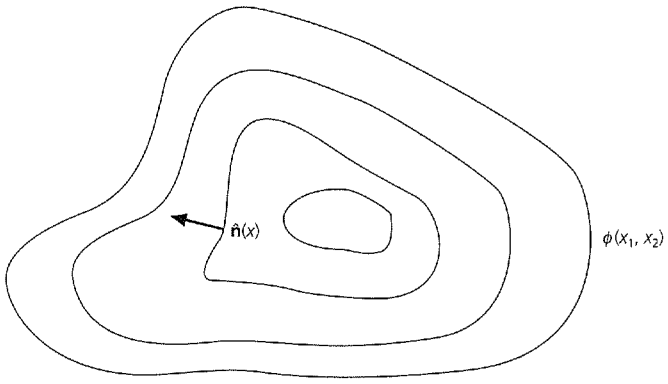


Fig. A.6-1 A scalar field demonstrating the concept of a gradient. If $\phi(x_1, x_2)$ gives the elevation, the gradient can be used to find the slope in the $\hat{\mathbf{n}}$ direction at a point (x_1, x_2).

gives the *directional derivative* in the $\hat{\mathbf{n}}$ direction. Because both $\hat{\mathbf{n}}$ and $\nabla\phi$ are functions of position, the directional derivative varies in space. At any point, the maximum value of the scalar product occurs for $\hat{\mathbf{n}}$ parallel to the gradient, so the gradient points in the direction of the steepest slope along which ϕ changes most rapidly. The scalar product is zero when $\hat{\mathbf{n}}$ is perpendicular to the gradient, so the gradient is perpendicular to curves of constant ϕ. These concepts are also used in three dimensions.

In index notation, the gradient is written as

$$(\nabla\phi)_i = \frac{\partial\phi}{\partial x_i} = \phi_{,i}, \qquad (6)$$

where the last form uses a common (if sometimes confusing) notation in which differentiation is indicated by a comma. The notation, with one free index, shows that the gradient is a vector. By contrast, the directional derivative, written as

$$\hat{\mathbf{n}} \cdot \nabla\phi = n_i\frac{\partial\phi}{\partial x_i} = n_i\phi_{,i}, \qquad (7)$$

has an implied sum over i and is a scalar.

Often, the gradients of quantities are important physically because an effect depends on spatial variations of a field. For example, the flow of heat depends on the gradient of the temperature field (Sections 5.3.2, 5.4.1), and the gradient of the pressure field in the atmosphere is important for the weather.

A.6.3 Divergence

A related operation that describes the spatial variation of a vector field is the *divergence*. The divergence of a vector field $\mathbf{u}(\mathbf{x})$ is given by the scalar product of the ∇ operator with $\mathbf{u}(\mathbf{x})$ as

$$\text{div}\,\mathbf{u} = \nabla \cdot \mathbf{u} = \frac{\partial u_1}{\partial x_1} + \frac{\partial u_2}{\partial x_2} + \frac{\partial u_3}{\partial x_3}, \qquad (8)$$

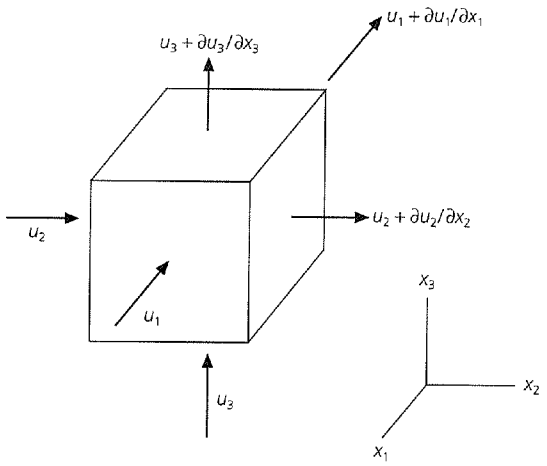


Fig. A.6-2 The divergence, formed from the differences between the flow into one face of a volume and the flow out of the opposite face, gives the net flow through a unit volume.

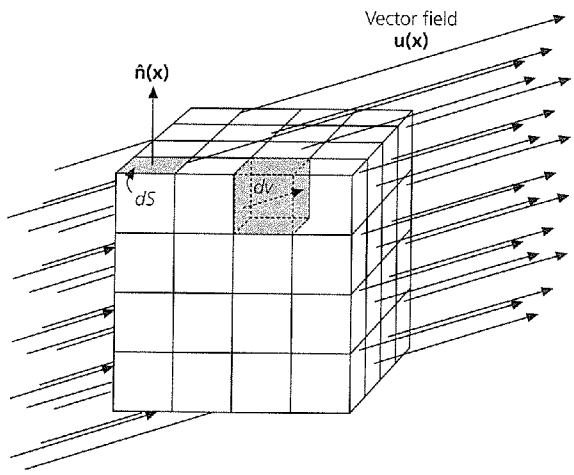


Fig. A.6-3 Geometry for the divergence theorem: $\hat{\mathbf{n}}(\mathbf{x})$ is a unit vector pointing outward at the point $\mathbf{x}$ from an element dS of the surface S that encloses a volume dV.

which yields a scalar field because the vector components and their derivatives are functions of position.

The divergence frequently arises in conservation equations. For example, if $\mathbf{u}(\mathbf{x})$ is the velocity as a function of position in a fluid, $\nabla \cdot \mathbf{u}(\mathbf{x})$ gives the net outflow of material per unit time from a unit volume at position $\mathbf{x}$ (Fig. A.6-2). To see this, note that, to first order, the net flow in the x_2 direction is the difference between the flow out the far side, $u_2 + \partial u_2/\partial x_2$, and that into the near side, u_2, given as

$$u_2 + \frac{\partial u_2}{\partial x_2} - u_2 = \frac{\partial u_2}{\partial x_2}. \tag{9}$$

Adding similar terms for the net flow in the x_1 and x_3 directions gives the divergence (Eqn 8). If the divergence is positive, there is a net outward flow, whereas a negative divergence indicates a net inflow.

This idea can be applied to any vector field $\mathbf{u}(\mathbf{x})$. Consider the problem of finding the net outflow from a region with volume V and surface S. If $\hat{\mathbf{n}}(\mathbf{x})$ is the unit normal vector pointing outward at a point $\mathbf{x}$ on the surface (Fig. A.6-3), the scalar product $\hat{\mathbf{n}}(\mathbf{x}) \cdot \mathbf{u}(\mathbf{x})$ gives the outward *flux* per unit area at that point. Integrating the flux over the surface then gives the total flux. Another way to compute the total flux is to integrate the divergence over the volume. These two methods give the same flux, so

$$\int_S \hat{\mathbf{n}} \cdot \mathbf{u} dS = \int_V \nabla \cdot \mathbf{u} dV. \tag{10}$$

This relation, *Gauss's theorem*, or the *divergence theorem*, says that what accumulates inside a volume is determined by the integral over its surface of what goes out. If we think of the volume as many adjacent cells, the flow out of one cell is the flow into an adjacent cell, which cancels to zero. Only flow in or out of the volume's surface is not canceled out in this way. Written in full, $\int dV$ is a triple integral over the volume, and $\int dS$ is a double integral over the surface.

In index notation, using the summation convention, the divergence is written

$$\nabla \cdot \mathbf{u} = \frac{\partial u_i}{\partial x_i} = u_{i,i}, \tag{11}$$

which is a scalar because no free index remains. Gauss's theorem is written

$$\int_S u_i n_i dS = \int_V \frac{\partial u_i}{\partial x_i} dV, \tag{12}$$

or, using the comma notation for derivatives,

$$\int_S u_i n_i dS = \int_V u_{i,i} dV. \tag{13}$$

As before, it is implicit in the notation that the field $\mathbf{u}$, its derivatives, and the normal vector $\hat{\mathbf{n}}$ vary with position.

A.6.4 Curl

The *curl* operator, the cross product of the ∇ operator with a vector field, yields another vector field

$$\nabla \times \mathbf{u} = \hat{\mathbf{e}}_1\left(\frac{\partial u_3}{\partial x_2} - \frac{\partial u_2}{\partial x_3}\right) + \hat{\mathbf{e}}_2\left(\frac{\partial u_1}{\partial x_3} - \frac{\partial u_3}{\partial x_1}\right) + \hat{\mathbf{e}}_3\left(\frac{\partial u_2}{\partial x_1} - \frac{\partial u_1}{\partial x_2}\right). \tag{14}$$

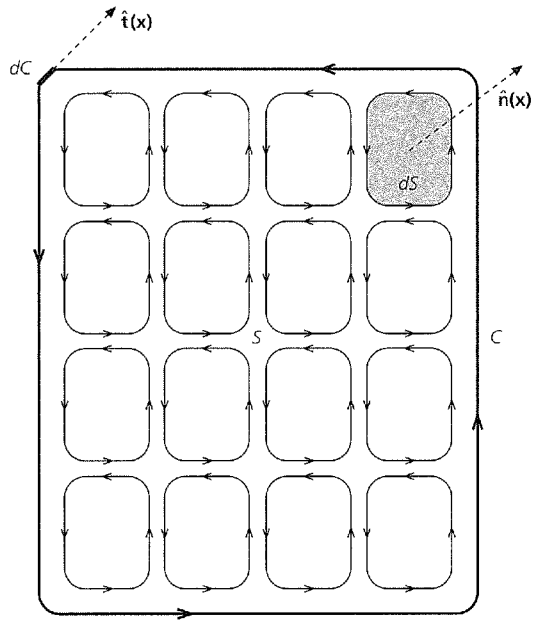


Fig. A.6-4 Geometry for Stokes' theorem: $\hat{\mathbf{n}}(\mathbf{x})$ is a unit vector pointing outward at the point $\mathbf{x}$ from an element dS of the surface S. dC is an element of the curve C bounding S, with tangent $\hat{\mathbf{t}}(\mathbf{x})$.

This can be written as a determinant

$$\nabla \times \mathbf{u} = \det \begin{pmatrix} \hat{\mathbf{e}}_1 & \hat{\mathbf{e}}_2 & \hat{\mathbf{e}}_3 \\ \dfrac{\partial}{\partial x_1} & \dfrac{\partial}{\partial x_2} & \dfrac{\partial}{\partial x_3} \\ u_1 & u_2 & u_3 \end{pmatrix}, \tag{15}$$

or, using index notation, in a compact form as

$$\nabla \times \mathbf{u} = \varepsilon_{ijk} \frac{\partial u_k}{\partial x_j} = \varepsilon_{ijk} u_{k,j}. \tag{16}$$

Some physical insight into the curl comes from *Stokes' theorem*, which relates the integral of the curl of a vector field over a surface S to the line integral around a curve C bounding S (Fig. A.6-4) as

$$\int_C \mathbf{u} \cdot \hat{\mathbf{t}}\, dC = \int_S (\nabla \times \mathbf{u}) \cdot \hat{\mathbf{n}}\, dS. \tag{17}$$

Here dS is an element of surface area with normal $\hat{\mathbf{n}}(\mathbf{x})$, and dC is an element of the curve with tangent $\hat{\mathbf{t}}(\mathbf{x})$. Analogous to the case of Gauss's theorem applied to a volume, we can think of the surface as composed of infinitesimal tiles, each with a line integral of $\mathbf{u} \cdot \hat{\mathbf{t}}$ around it. The border of each tile is shared with another tile, but, because the line integral, or *circulation*, is computed in a counterclockwise manner, the integrals along this border are the same but of opposite sign for the two tiles, and therefore cancel. The segments of the line integrals cancel between all the tiles except those on the outer border that have no adjacent circulation to cancel them.

If the line integral is nonzero, the vector field has a net rotation along the curve, so the integral of its curl over the surface is nonzero. The curl of a vector field shows where rotations arise. A common application is describing the velocity field of a moving fluid. The upper portion of Fig. A.6-5 shows streamlines, lines parallel to the velocity vector at any point, for a viscous fluid flowing past a circular object. The velocity is zero at the object, and increases with distance away from it. The flow is symmetric on the bottom of the object. The lower portion of the figure shows contours of the curl of the velocity field with larger values, indicating greater rotations, close to the object.

Two useful identities, whose proofs are left for the problems, are that the curl of a gradient and the divergence of a curl are zero:

$$\nabla \cdot (\nabla \times \mathbf{u}) = 0 \tag{18}$$

$$\nabla \times (\nabla \phi) = 0. \tag{19}$$

Equation 19 can be used with Stokes' theorem to show that for a vector field written as the gradient of a scalar, the curl, and hence circulation around an arbitrary curve, are zero. This idea is used in mechanics to prove that a conservative force (one that can be written as the gradient of a potential) has a line integral that is independent of path, because its circulation around any path is zero. These relations give insight into seismic waves, because P waves have no curl and S waves have no divergence (Section 2.4.1).

A.6.5 Laplacian

The *Laplacian* operator is formed by taking the divergence of the gradient of a scalar field, which yields a scalar field

$$\nabla^2 \phi = \nabla \cdot \nabla \phi = \frac{\partial^2 \phi}{\partial x_1^2} + \frac{\partial^2 \phi}{\partial x_2^2} + \frac{\partial^2 \phi}{\partial x_3^2} = \phi_{,ii}, \tag{20}$$

where the last form uses index notation and the summation convention. By analogy, the Laplacian of a vector field is a vector field whose components in Cartesian coordinates are the Laplacians of the original vector components,

$$\nabla^2 \mathbf{u} = (\nabla^2 u_1, \nabla^2 u_2, \nabla^2 u_3). \tag{21}$$

For example, the $\hat{\mathbf{e}}_1$ component of $\nabla^2 \mathbf{u}$ is

$$\frac{\partial^2 u_1}{\partial x_1^2} + \frac{\partial^2 u_1}{\partial x_2^2} + \frac{\partial^2 u_1}{\partial x_3^2}. \tag{22}$$

In Cartesian coordinates, the Laplacian of a vector satisfies

$$\nabla^2 \mathbf{u} = \nabla(\nabla \cdot \mathbf{u}) - \nabla \times (\nabla \times \mathbf{u}), \tag{23}$$

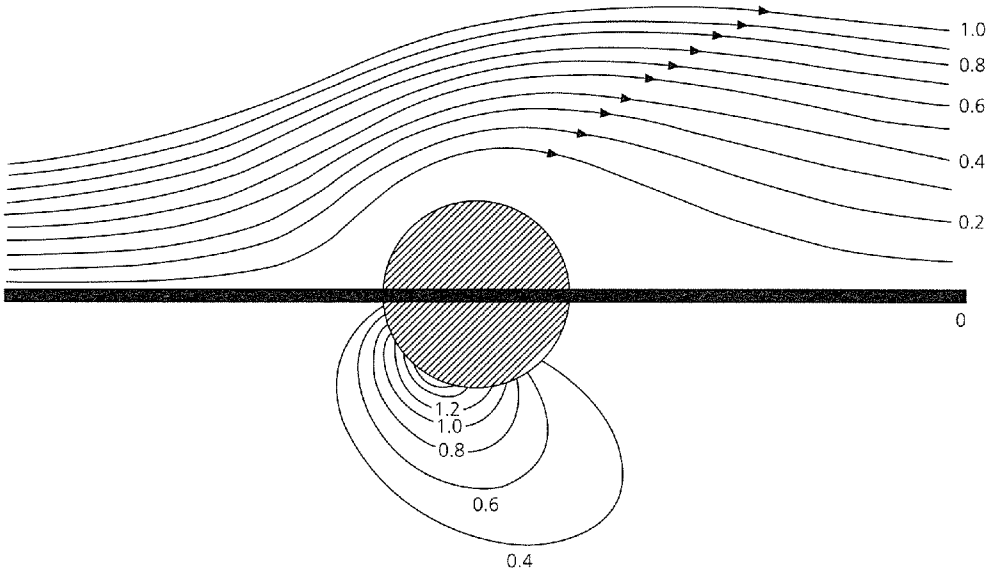


Fig. A.6-5 *Top*: streamlines showing the velocity of fluid flow around an object. Numbers on streamlines show the magnitude of the velocity. *Bottom*: contours of the curl for this velocity field. The curl is greatest near the sphere, where the fluid flow lines are the most curved. (After Batchelor, 1967. Reprinted with the permission of Cambridge University Press.)

an obscure-looking relation that is useful in deriving the existence of *P* and *S* waves.

A.7 Spherical coordinates

The vector operations discussed so far were performed in Cartesian coordinates, in which the unit basis vectors $(\hat{e}_1, \hat{e}_2, \hat{e}_3)$ point in the same direction everywhere. There are, however, situations in which non-Cartesian coordinate systems without these nice properties are useful. In particular, *spherical* coordinates often simplify the solution of problems with a high degree of symmetry about a point.

A.7.1 The spherical coordinate system

In a spherical coordinate system, a point defined by a position vector **x** is described by its radial distance from the origin, $r = |\mathbf{x}|$, and two angles. θ is the *colatitude*, or angle between **x** and the x_3 axis, and ϕ, the *longitude*, is measured in the x_1–x_2 plane. Often the *latitude*, $90° - \theta$, is used instead of the colatitude. Spherical coordinates are often used in seismology because the earth is approximately spherically symmetric, varying with depth much more than laterally. Thus properties like velocity and density are often approximated as functions only of r, independent of θ and ϕ.

Figure A.7-1 shows the relations between rectangular and spherical coordinates. If the vector **x** is written as

$$\mathbf{x} = x_1\hat{e}_1 + x_2\hat{e}_2 + x_3\hat{e}_3, \tag{1}$$

then its components in rectangular coordinates (x_1, x_2, x_3) are described by spherical coordinates as

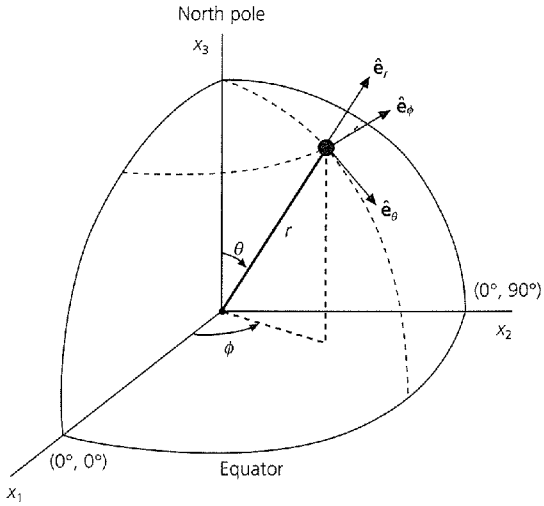


Fig. A.7-1 Relations between spherical (r, θ, ϕ) and Cartesian coordinates (x_1, x_2, x_3). (After Marion, 1970. From *Classical Dynamics of Particles and Systems*, 2nd edn, copyright 1970 by Academic Press, reproduced by permission of the publisher.)

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} r\sin\theta\cos\phi \\ r\sin\theta\sin\phi \\ r\cos\theta \end{pmatrix}. \tag{2}$$

Conversely, the spherical coordinates r, θ, and ϕ can be written as

$$r = (x_1^2 + x_2^2 + x_3^2)^{1/2}, \quad \theta = \cos^{-1}(x_3/r), \quad \phi = \tan^{-1}(x_2/x_1). \tag{3}$$

In the equatorial $(x_1$–$x_2)$ plane, $\theta = 90°$, $\cos\theta = 0$, $\sin\theta = 1$, so $x_1 = r\cos\phi$, $x_2 = r\sin\phi$, and $x_3 = 0$. This is the same as the polar

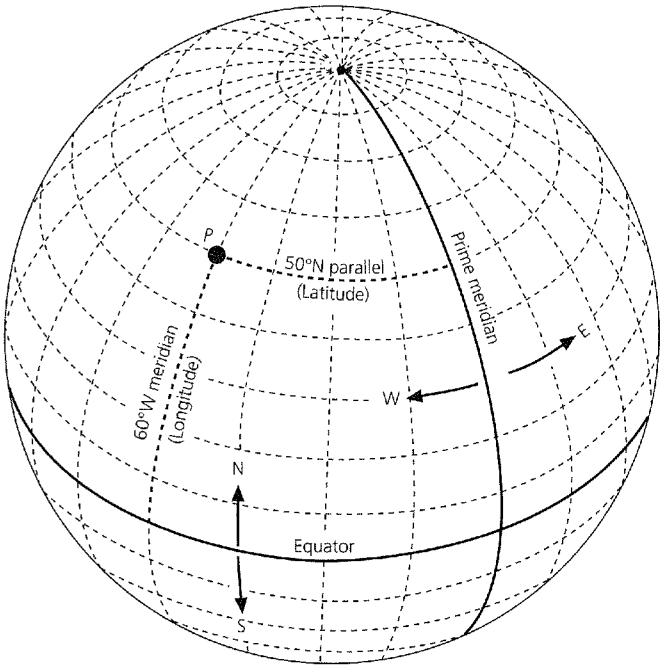


Fig. A.7-2 Geometry of the latitude and longitude system used to locate points on the earth's surface. A point P at 50°N, 60°W ($\theta = 40°$, $\phi = -60°$) is shown. (After Strahler, 1969.)

coordinate system described in Section A.3.1. Along the x_3 axis we have $\theta = 0°$, so $x_1 = x_2 = 0$, and $x_3 = r$. Any of these expressions written in terms of colatitude θ can be converted to latitude $\lambda = 90° - \theta$, using $\cos\theta = \sin\lambda$ and $\sin\theta = \cos\lambda$.

This coordinate system is the familiar one (Fig. A.7-2) used to locate points within the earth or on its surface, $r = a$. For this purpose, the origin is placed at the center of the earth, and the x_3 axis is defined by a line from the center of the earth through the north pole. The intersections of planes containing the x_3 axis with the earth's surface define *meridians*, lines of constant longitude. The x_1 axis intersects the equator at the *prime meridian*, on which ϕ is defined as zero, which has been chosen to run through Greenwich, England. The intersection of planes perpendicular to the x_3 axis with the earth's surface define *parallels*, lines of constant colatitude or latitude. Meridians are a special case of *great circles*, lines on the surface defined by the intersection of a plane through the origin with the surface of the spherical earth. Parallels are a special case of *small circles*, which are lines on the surface defined by the intersection of the surface of the spherical earth with a plane normal to a radius vector.

These conventions allow the colatitude θ ($0° \le \theta < 180°$) and longitude ϕ ($0° \le \phi < 360°$) to define a unique point on the earth's surface. Often locations are described in terms of latitudes north and south of the equator, and longitudes east and west of Greenwich. North and south latitudes correspond, respectively, to colatitudes less than or greater than 90°. Because ϕ measures longitude east of the prime meridian, west longitudes correspond to values of ϕ less than 0° or greater than 180°. Thus a point at (10°S, 110°W) has $\theta = 90° + 10° = 100°$, and $\phi = -110° = 360° - 110° = 250°$.

At any point, unit spherical basis vectors ($\hat{e}_r$, $\hat{e}_\theta$, $\hat{e}_\phi$) can be defined in the direction of increasing r, θ, and ϕ. $\hat{e}_r$ points away from the origin, and gives the upward vertical direction. $\hat{e}_\theta$ points south, and $\hat{e}_\phi$ points east. These two are sometimes written in terms of north- and east-pointing unit vectors, $\hat{e}_{NS} = -\hat{e}_\theta$ and $\hat{e}_{EW} = \hat{e}_\phi$.

An important feature of the unit spherical basis vectors is that at different points they are oriented differently with respect to the Cartesian axes. The Cartesian unit basis vectors ($\hat{e}_1$, $\hat{e}_2$, $\hat{e}_3$) point in the same direction everywhere. By contrast, for example, $\hat{e}_r$ points in the $\hat{e}_3$ direction at the north pole, and in the $-\hat{e}_3$ direction at the south pole. This effect is described by the Cartesian ($\hat{e}_1$, $\hat{e}_2$, $\hat{e}_3$) components of the unit spherical basis vectors, at a point with colatitude θ and longitude ϕ:

$$\hat{e}_\phi = \begin{pmatrix} -\sin\phi \\ \cos\phi \\ 0 \end{pmatrix}, \quad \hat{e}_\theta = \begin{pmatrix} \cos\theta\cos\phi \\ \cos\theta\sin\phi \\ -\sin\theta \end{pmatrix}, \quad \hat{e}_r = \begin{pmatrix} \sin\theta\cos\phi \\ \sin\theta\sin\phi \\ \cos\theta \end{pmatrix}. \quad (4)$$

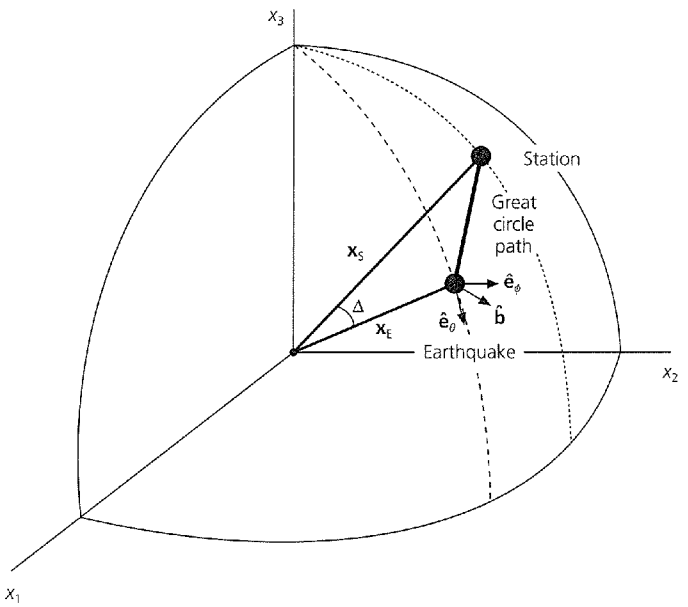
The dependence on the colatitude and longitude describes how the orientation with respect to the Cartesian axes changes.

At any point, the spherical basis vectors ($\hat{e}_r$, $\hat{e}_\theta$, $\hat{e}_\phi$) form an orthonormal set. For problems whose spatial extent is small enough that the curvature of the earth can be ignored, these basis vectors provide a useful local coordinate system.

A.7.2 *Distance and azimuth*

Spherical coordinates are especially useful in describing the geographic relation between two points on the earth's surface. A common application is to find the distance between points and the direction of the great circle arc joining them. A great circle arc is the shortest path between points on a sphere, so if seismic velocity varies only with depth, the fastest path along the surface is the great circle arc, and the fastest paths through the interior are in the plane of the great circle and the center of the earth. Because velocities vary laterally by only a few percent throughout most of the earth (and imperceptibly in the liquid outer core), this is a good approximation for most seismic applications. The source-to-receiver distance is often given in terms of the angle Δ subtended at the center of the earth by the great circle arc between the two points (Fig. A.7-3). If Δ is expressed in radians, then the length s (in km) of the arc along the earth's surface is $R\Delta$, where R is the earth's radius ($\approx$ 6371 km). If Δ is expressed in degrees, $s = R\Delta\pi/180$, so one degree of arc equals 111.2 km.

Consider the great circle arc connecting an earthquake whose epicenter is at (θ_E, ϕ_E) and a seismic station at (θ_S, ϕ_S). Seismic waves that traveled along the great circle arc (or in the plane of this arc and the center of the earth) left the earthquake in a direction given by the *azimuth* angle ζ measured clockwise from the local direction of north at the epicenter to the great



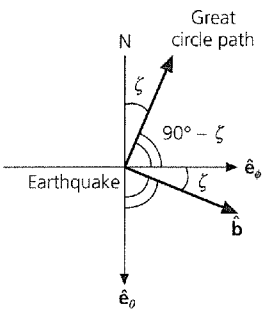


Fig. A.7-3 Geometry of the great circle path between an earthquake epicenter and seismic station (*left*), showing the convention for defining the azimuth, ζ (*right*).

circle arc. These waves arrive at the seismometer from a direction described by the *back azimuth* angle ζ' measured clockwise from the local direction of north at the seismometer to the great circle arc. To find these quantities, the Cartesian components of the position vectors for the earthquake and the station are written, using Eqn 2:

$$\mathbf{x}_E = \begin{pmatrix} R \sin\theta_E \cos\phi_E \\ R \sin\theta_E \sin\phi_E \\ R \cos\phi_E \end{pmatrix} \quad \mathbf{x}_S = \begin{pmatrix} R \sin\theta_S \cos\phi_S \\ R \sin\theta_S \sin\phi_S \\ R \cos\theta_S \end{pmatrix}. \tag{5}$$

The distance Δ, the angle between $\mathbf{x}_S$ and $\mathbf{x}_E$, is given by the scalar product

$$\mathbf{x}_S \cdot \mathbf{x}_E = R^2 \cos\Delta, \tag{6}$$

so

$$\Delta = \cos^{-1}[\cos\theta_E \cos\theta_S + \sin\theta_E \sin\theta_S \cos(\phi_S - \phi_E)]. \tag{7}$$

This formula defines Δ uniquely between 0 and 180°. This shorter portion of the great circle is called the *minor arc* connecting the two points; the longer portion, known as the *major arc*, is $(360° - \Delta)$ degrees long.

To compute the azimuth from the earthquake to the station, consider $\hat{\mathbf{b}}$, a unit vector normal to the great circle in the local horizontal plane at $\mathbf{x}_E$, which is written using the vector product of the position vectors

$$\mathbf{x}_S \times \mathbf{x}_E = \hat{\mathbf{b}} R^2 \sin\Delta. \tag{8}$$

Evaluation of the vector product gives

$$\hat{\mathbf{b}} = \frac{1}{\sin\Delta} \begin{pmatrix} \sin\theta_S \cos\theta_E \sin\phi_S - \sin\theta_E \cos\theta_S \sin\phi_E \\ \cos\theta_S \sin\theta_E \cos\phi_E - \cos\theta_E \sin\theta_S \cos\phi_S \\ \sin\theta_S \sin\theta_E \sin(\phi_E - \phi_S) \end{pmatrix}. \tag{9}$$

The azimuth angle ζ, measured clockwise from north, is then given (Fig. A.7-3) by

$$\cos\zeta = \hat{\mathbf{b}} \cdot \hat{\mathbf{e}}_\phi = \frac{1}{\sin\Delta}(\cos\theta_S \sin\theta_E - \sin\theta_S \cos\theta_E \cos(\phi_S - \phi_E)) \tag{10}$$

and

$$\sin\zeta = \hat{\mathbf{b}} \cdot \hat{\mathbf{e}}_\theta = \frac{1}{\sin\Delta} \sin\theta_S \sin(\phi_S - \phi_E). \tag{11}$$

Use of both $\sin\zeta$ and $\cos\zeta$ makes the angle ζ unambiguous ($0° \le \zeta < 360°$). The azimuth from an earthquake to a receiver is useful, because earthquakes radiate more energy in some directions than in others (Chapter 4), so measurements at different azimuths yield information about the source.

The back azimuth ζ', obtained by reversing the indices E and S in Eqns 10 and 11, shows the direction from which seismic energy arrives at a seismometer. Seismometers typically record the north–south and east–west components of horizontal ground motion. Using the back azimuth, these observations can be converted into *radial* (along the great circle path) and *transverse* (perpendicular to the great circle path) components by a vector transformation (Eqn A.5.9). This distinction is made because waves appearing on these components propagated differently (Section 2.4). The azimuth and back azimuth

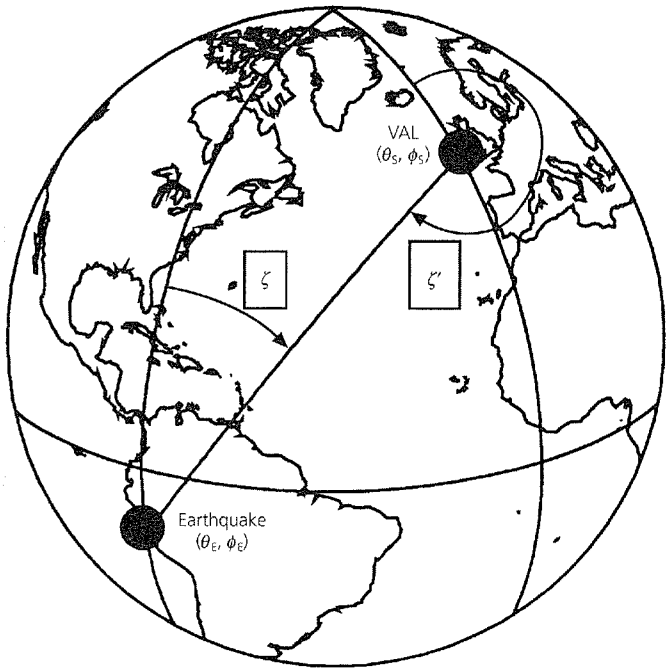


Fig. A.7-4 Geometry of the great circle path for an earthquake in the Peru trench recorded at station VAL (Valentia, Ireland). The azimuth, ζ, and back azimuth, ζ', are not simply related, due to the sphericity of the earth.

angles are measured clockwise from north, a geographic convention which contrasts with the mathematical one of measuring angles counterclockwise from the x_1 direction. Figure A.7-4 illustrates this geometry for an earthquake in the Peru trench ($\theta_E = 102°$, $\phi_E = -78°$) recorded at station VAL (Valentia, Ireland; $\theta_S = 38°$, $\phi_S = -10.25°$). The resulting distances and azimuths are $\Delta = 86°$, $\zeta = 35°$, $\zeta' = 245°$.[1]

This analysis assumes that the earth is perfectly spherical. In fact, the earth is flattened by its rotation into a shape close to an oblate ellipsoid, so the radius varies with colatitude approximately as

$$r(\theta) = R_e(1 - f\cos^2\theta), \tag{12}$$

where R_e is the equatorial radius, 6378 km. The flattening factor f is approximately 3.35×10^{-3}, or about 1/298, so the polar radius R_p is 6357 km. An average radius can be defined as the radius of a sphere with the same volume as the earth, if it were a perfect ellipsoid. Because the volume of an ellipsoidal earth would be $(4/3)\pi R_e^2 R_p$, and a sphere of radius R has volume $(4/3)\pi R^3$, the average radius is 6371 km. For certain applications the ellipticity is included in precise distance calculations.

[1] These distance–azimuth equations also have nonseismological applications because ships and aircraft follow the shortest (great circle) paths between two points when possible.

A.7.3 Choice of axes

Spherical coordinates are also used with axes different from the geographic ones. Because the physics of a problem does not depend on the choice of coordinates, a set of coordinates that simplifies the relevant expressions is used. For example, in earthquake source studies, the x_3 axis can be chosen to go from the center of the earth to the location of the earthquake. The prime meridian, and hence x_1, axis can be selected so that the fault is oriented in the direction $\phi = 0$. These axes simplify the description of the seismic waves radiated by an earthquake, because the distance Δ from the source is now the colatitude. Moreover, the radiation pattern generally has a high degree of symmetry about the fault, so simple functions of ϕ appear. By contrast, the radiation pattern need have no symmetry about the North pole and Greenwich meridian, so a description in those coordinates would usually be more complicated.

Fortunately, a coordinate system referred to the earthquake location does not make describing the propagation of waves from the source any more difficult. Because earth structure varies primarily with depth, the spherical symmetry about the center of the earth is independent of the axis orientation chosen. The geographical convention in which the earth rotates about the x_3 axis is helpful for navigation. In most seismological applications, however, the north direction has no particular significance because the propagation of seismic waves is essentially unaffected by the earth's rotation. The choice of a prime meridian is arbitrary; in the early nineteenth century some American maps had it through Washington DC, and some French maps had it through Paris.

A.7.4 Vector operators in spherical coordinates

Because at a point on the sphere the unit spherical basis vectors are oriented up, south, and east, the basis vectors at different locations are generally not parallel. This makes the vector differential operators more complicated, because these operators involve taking spatial derivatives of vectors. In Cartesian coordinates the unit basis vectors are not affected by this differentiation because they do not change orientation, so only derivatives of the components need be taken. In spherical coordinates, because a vector **u** is

$$\mathbf{u} = u_r\hat{\mathbf{e}}_r + u_\theta\hat{\mathbf{e}}_\theta + u_\phi\hat{\mathbf{e}}_\phi, \tag{13}$$

differential operators acting on **u** must incorporate the derivatives of the basis vectors. Thus, in spherical coordinates, for a scalar field ψ and a vector field **u**:

$$\text{grad}\,\psi = \hat{\mathbf{e}}_r\frac{\partial\psi}{\partial r} + \hat{\mathbf{e}}_\theta\frac{1}{r}\frac{\partial\psi}{\partial\theta} + \hat{\mathbf{e}}_\phi\frac{1}{r\sin\theta}\frac{\partial\psi}{\partial\phi} \tag{14}$$

$$\text{div}\,\mathbf{u} = \frac{1}{r^2}\frac{\partial}{\partial r}(r^2u_r) + \frac{1}{r\sin\theta}\frac{\partial}{\partial\theta}(\sin\theta\, u_\theta) + \frac{1}{r\sin\theta}\frac{\partial u_\phi}{\partial\phi} \tag{15}$$

$$\text{curl } \mathbf{u} = \hat{\mathbf{e}}_r \frac{1}{r \sin\theta}\left(\frac{\partial}{\partial\theta}(\sin\theta\, u_\phi) - \frac{\partial u_\theta}{\partial\phi}\right) + \hat{\mathbf{e}}_\theta \frac{1}{r\sin\theta}\left(\frac{\partial u_r}{\partial\phi} - \sin\theta\frac{\partial}{\partial r}(r u_\phi)\right) + \hat{\mathbf{e}}_\phi \frac{1}{r}\left(\frac{\partial}{\partial r}(r u_\theta) - \frac{\partial u_r}{\partial\theta}\right) \tag{16}$$

$$\nabla^2\psi = \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial\psi}{\partial r}\right) + \frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial\psi}{\partial\theta}\right) + \frac{1}{r^2\sin^2\theta}\frac{\partial^2\psi}{\partial\phi^2}. \tag{17}$$

These expressions are used when we discuss spherical waves in Section 2.4 and the earth's normal modes in Section 2.9.

A final point worth noting is that the elements of volume and surface used in integrals are different in spherical coordinates from rectangular coordinates. In spherical coordinates (Fig. A.7-5) there are several scale factors, so an element of surface area is

$$dS = r^2 \sin\theta\, d\theta d\phi, \tag{18}$$

and an element of volume is

$$dV = r^2 \sin\theta dr\, d\theta d\phi. \tag{19}$$

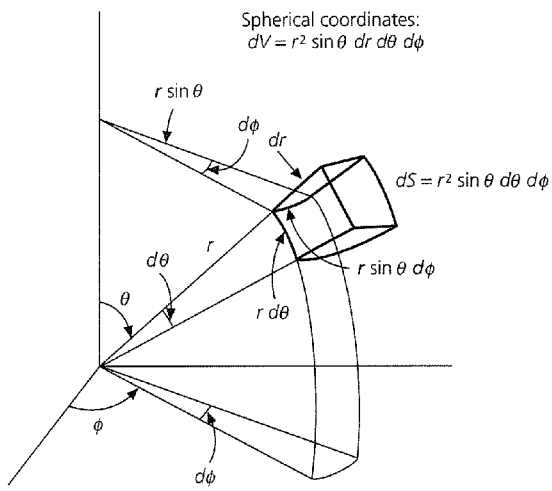


Fig. A.7-5 Definition of the element of volume in spherical coordinates. Unlike the case of Cartesian coordinates, the volume element in spherical coordinates in not a cube. (Marion, 1970. From *Classical Dynamics of Particles and Systems*, 2nd edn, copyright 1970 by Academic Press, reproduced by permission of the publisher.)

A.8 Scientific programming

Most seismological applications require computers, and these requirements, especially in exploration applications with very large data volumes, have spurred the development of computer software and hardware. Some remarks about the use of computers in seismology thus seem appropriate.

Computer usage in seismology includes several broad and overlapping categories:

- Computers are often used in data acquisition and recording systems.
- Data are initially displayed and manipulated using computers.
- Subsequent analysis is frequently done using computers. For example, seismograms can be filtered to enhance certain frequencies or combined to better resolve certain features.
- Theoretical, or *synthetic*, seismograms are often computed for a range of the parameters under study and compared to data to find the best fit.
- Computers are used to *invert* seismological data to determine the parameters of a model which best matches the data.
- Computer modeling is often used to draw geological inferences from seismological observations. For example, seismic velocity data are compared to the predictions of models for the velocity of rock as a function of composition, temperature, and pressure.

These applications often require *scientific programming*, a programming style used for essentially mathematical applications. Some problems in this book also require scientific programming. Although programming is a matter of personal style, this section discusses several points that may be helpful. The suggested reading provides some starting points for readers interested in pursuing these topics further.

A.8.1 Example: synthetic seismogram calculation

Consider a program to compute a synthetic seismogram for waves in a one-dimensional constant-velocity medium, a mathematically idealized string that illustrates features of wave behavior. The program is based on $u(x, t)$, the displacement as a function of position x and time t. The displacement is zero at the fixed ends of the string, $x = 0$ and $x = L$, between which waves travel at speed v. As in Section 2.2.5, the displacement can be written as the sum of the normal modes of the string, each of which is a standing wave with n half wavelengths along the string,

$$u_n(x, t) = \sin(n\pi x/L)\cos(\omega_n t), \tag{1}$$

and vibrates at a characteristic frequency, or *eigenfrequency*,

$$\omega_n = n\pi v/L. \tag{2}$$

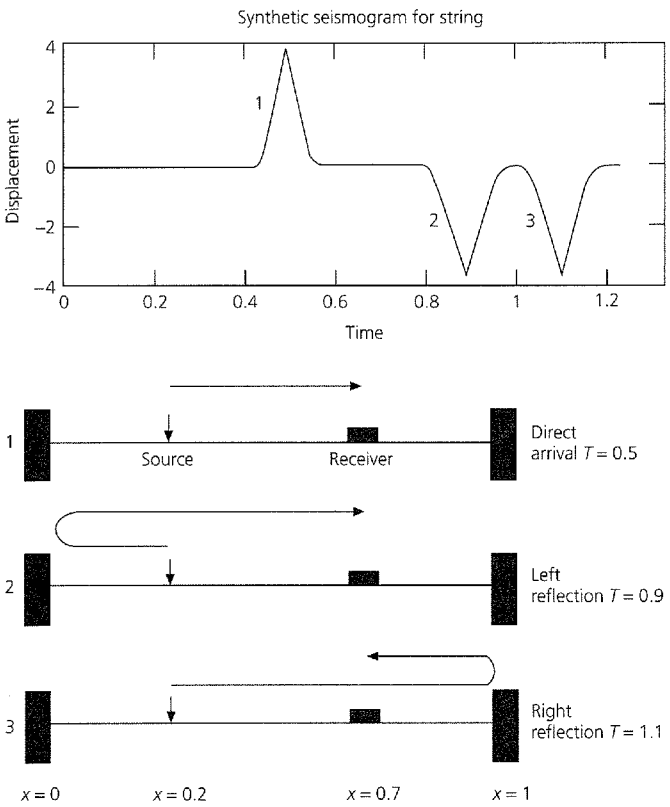


Fig. A.8-1 *Top*: Synthetic seismogram for a string showing the direct wave arrival (1) and reflections (2, 3) from both ends. *Bottom*: Geometry showing source and receiver positions, and the times of the direct and reflected arrivals.

If a source at position x_s generates a pulse at time zero with duration τ, the propagating waves are described by a weighted sum of the modes

$$u(x, t) = \sum_{n=1}^{\infty} \sin(n\pi x/L) \sin(n\pi x_s/L) \cos(\omega_n t) \exp[-(\omega_n \tau)^2/4]. \quad (3)$$

Given the displacement $u(x, t)$ for any position and time, a seismogram ("stringogram") giving the displacement versus time at a receiver position x_r is $u(x_r, t)$. Alternatively, a "snapshot" of the displacement everywhere on the string at time t_0 is $u(x, t_0)$.

Consider a program to evaluate a synthetic seismogram using this sum. For simplicity, we use a string of length 1 m[1] with a wave speed 1 m/s, a source at $x_s = 0.2$ m and a receiver at $x_r = 0.7$ m. To approximate the infinite sum, the program adds up 200 modes. The seismogram (Fig. A.8-1, *top*) is calculated at 50 time steps, covering 1.25 s. This program is written in Fortran, a language that is especially suitable for scientific programming and is therefore commonly used in seismology (and thus in this book). The program could be also written in other languages, but the general points would still apply.

```
C SYNTHETIC SEISMOGRAM FOR HOMOGENEOUS STRING
C DISPLACEMENT U AS FUNCTION OF TIME T
C CALCULATED BY NORMAL MODE SUMMATION
      DIMENSION U(200)
      PI = 3.1415927
C
C PARAMETERS (NORMALLY WOULD COME FROM INPUT)
C STRING LENGTH (M)
      ALNGTH = 1.0
C VELOCITY (M/S)
      C = 1.0
C NUMBER OF MODES
      NMODE = 200
C SOURCE POSITION (M)
      XSRC = 0.2
C RECEIVER POSITION (M)
      XRCVR = 0.7
C SEISMOGRAM TIME DURATION (S)
      TDURAT = 1.25
C NUMBER TIME STEPS
      NTSTEP = 50
C TIME STEP (S)
      DT = TDURAT/NTSTEP
C SOURCE SHAPE TERM
      TAU = .02
C
C LIST PARAMETERS
      WRITE (6,3000)
3000  FORMAT('SYNTHETIC SEISMOGRAM FOR STRING')
      WRITE (6,3001) NMODE
3001  FORMAT('NUMBER OF MODES', I6)
      WRITE (6,3002) ALNGTH, C
3002  FORMAT ('LENGTH (M)' F7.3, 'VELOCITY,
     X (M/S)', F7.3)
      WRITE (6,3003) XSRC, XRCVR
3003  FORMAT ('POSITION (M): SOURCE', F7.3,
     X 'RECEIVER', F7.3)
      WRITE (6,3004) TDURAT, NTSTEP
3004  FORMAT ('SEISMOGRAM DURATION (S)', F7.3,
     X I6, 'TIME STEPS')
      WRITE (6,3005) TAU
3005  FORMAT ('SOURCE SHAPE TERM', F7.3)
C
C INITIALIZE DISPLACEMENT
        DO 5 I = 1, NTSTEP
      U(I) = 0.0
5     CONTINUE
C
C OUTER LOOP OVER MODES
      DO 10 N = 1, NMODE
        ANPIAL = N*PI/ALNGTH
```

[1] It is easy to use arbitrary values on a computer; we could also use 1 km or 1 furlong. Finding a physical 1 km string is another matter . . .

```
C SPACE TERMS: SOURCE AND RECEIVER
        SXS = SIN(ANPIAL*XSRC)
        SXR = SIN(ANPIAL*XRCVR)
C MODE FREQUENCY
        WN = N*PI*C/ALNGTH
C TIME INDEPENDENT TERMS
        DMP = (TAU*WN)**2
        SCALE = EXP(-DMP/4.)
        SPACE = SXS*SXR*SCALE
C
C INNER LOOP OVER TIME STEPS
        DO 15 J = 1, NTSTEP
          T = DT*(J - 1)
          CWT = COS(WN*T)
C COMPUTE DISPLACEMENT
          U(J) = U(J) + CWT*SPACE
15      CONTINUE
10    CONTINUE
C
C OUTPUT SEISMOGRAM FOR LATER PLOTTING
      WRITE (6, 3101)(U(J), J = 1, NTSTEP)
3101  FORMAT (7F10.4)
      STOP
      END
```

This example brings out several points:

- *Is the answer correct?* Two different types of error occur in scientific programs. First, the *program* may be wrong. In this case, the mathematical formulation correctly describes the physical problem, but the program incorrectly implements this formulation. This is the usual situation, in which "bugs" are identified and corrected. Second, the *formulation* may be wrong, so the program correctly implements an incorrect mathematical model. This could occur because of a mathematical error, like an attempt to sum a divergent series, or a physical error, such as an equation that does not correctly describe waves on a string. An incorrect formulation is particularly disturbing because it cannot be detected by checking the program. For example, Fig. A.8-2 shows two computer simulations for waves bending as they pass from one medium into another with higher velocities. Figure A.8-2 (*top*) uses the correct formulation of Snell's law (Section 2.5), whereas Fig. A.8-2 (*bottom*) looks equally convincing but is wrong because the equation which the program illustrates is incorrect.

Programmers check for both types of errors by choosing cases for which the results can be predicted analytically and comparing the results to those of the program. Several tests are easily done for the string. The wave following the shortest (direct) path appears at the expected time, 0.5 s (Fig. A.8-1, *bottom*), because the source and the receiver are 0.5 m apart. The next two arrivals, reflections from the ends of the string, also occur at the expected times. Moreover, these arrivals have polarities opposite that of the initial pulse, as should occur (Section 2.2.3) upon reflection at the string's fixed ends. The program can also be checked for different string lengths, speeds, and source and receiver positions. Similarly, in addition to synthetic seismograms, displacements along the string at fixed times could be computed. Such tests are important, because if the mathematical model is not appropriate for the physical situation, then time spent debugging, documenting, and optimizing the program is wasted.

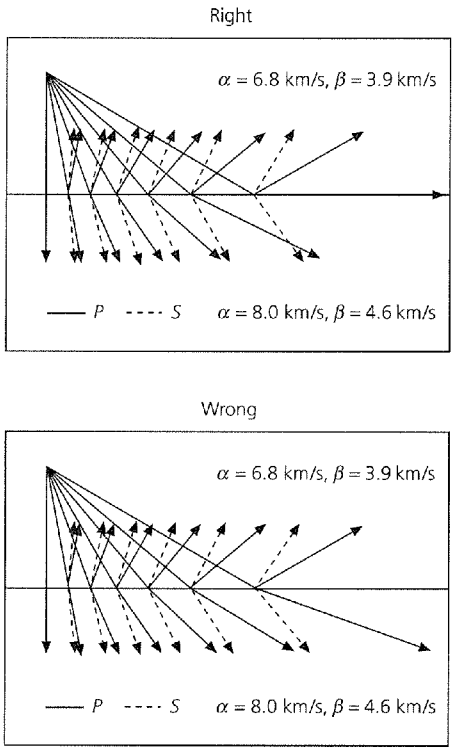


Fig. A.8-2 Demonstration of the danger that a program accurately computes an incorrect mathematical formulation. *Top*: A correct simulation of wave refraction using Snell's law, $\sin i_1/v_1 = \sin i_2/v_2$. *Bottom*: The same simulation using a wrong formula for Snell's law, $i_1/v_2 = i_2/v_2$.

- *The program is reasonably comprehensible.* Several features help clarify the program. The program's purpose and method are stated. Variable names somewhat resemble those in the equation: "SXS" is $\sin x_s$, and so on. Comments identify the functions of portions of the program.

- *The program uses loops and arrays.* The seismogram is described by the array U(J), and its values at successive times are calculated by looping. Using an array, rather than discrete variables UT1, UT2, etc., makes the program clearer, closer to the mathematical formulation, and simplifies output. The loop structure also makes the program clearer and allows the number of time steps to be changed simply by changing the parameter NTSTEP. Similarly, the number of modes is easily changed.

- *The output is labeled.* The seismogram was placed in an output file for later plotting. The parameters used to compute the seismogram are included, so examination of the output

```
C OUTER LOOP OVER MODES
      DO 10 N = 1, NMODE

          terms for each mode
          that do not depend on time

C INNER LOOP OVER TIME STEPS
          DO 15 J = 1, NTSTEP

          terms that depend on time

C COMPUTE DISPLACEMENT

15        CONTINUE
10    CONTINUE
```

Fig. A.8-3 Structure of the loops for the string synthetic seismogram calculation.

shows how it was computed. This helps avoid the common situation where, given a large collection of computer output, cases are rerun because it is unclear what parameters were used. Moreover, subsequent "improved" versions of the program can be checked to see whether they give the same results.

- *The program is somewhat efficient.* Some thought is generally put into *optimizing* scientific programs to make them run rapidly. The program could find the displacement by looping over time and summing all the modes at each time step. However, consideration of the equation shows that three terms, $\sin(n\pi x/l)$, $\sin(n\pi x_s/l)$, and $\exp[-(\omega_n\tau)^2/4]$ are evaluated only once for each mode, whereas only $\cos(\omega_n t)$ is evaluated for each time step. It is thus more efficient to loop over the modes and evaluate each at all times (Fig. A.8-3). Because the outer (mode) loop is executed 200 times, whereas the inner (time) loop is executed $200 \times 50 = 10{,}000$ times, the inner loop should be as efficient as possible. The program would run more slowly if the loops were reversed. The difference, though not significant for this calculation, might be significant for much larger numbers of time steps and modes.

Further improvements could be made to fully optimize the program. Optimization is not an end in itself, because the programmer's time and the intelligibility of the program are also important. Programmers typically try to write reasonably optimized programs without making them impossible to understand and debug. Once fully tested, a program that will be used heavily may be worth further optimization if the computer time savings justify the effort required. There is no point in "getting the wrong answer as fast as possible."[2] Certain computers, such as those using parallel processors, may require specialized optimization.

A.8.2 Programming style

The style in which programs are written can make them easier to develop, debug and use. A few suggestions, though not absolute rules, may be useful.

- *Document the program.* Computer programs can be almost useless without adequate documentation. Stonehenge has been described as "the world's largest undocumented computer system."[3] Failure to document is often justified by the assumption that the program will not be used again. This rationalization is self-fulfilling, because even the author may find an undocumented program difficult to reuse once the details are forgotten.

Documentation should state the program's goals and method. The input and output variables, their units, and how they are defined should be listed. Implicit assumptions and restrictions are worth noting. Comments should identify major portions of the program and describe their functions.

Documentation is best written when writing a program because it can aid in debugging. Moreover, once a program is fully written, it is harder to remember how it works. Documentation included in the program is less prone to be lost than that written separately.

Finally, documentation helps scientists exchange programs and work in collaboration. This can be useful, except in the apocryphal cases of programmers writing gigantic undocumented programs to maximize their job security.

- *Use modular programming.* Large programs can generally be divided into smaller subroutines or functions, which can be used like the functions (e.g., sine, square root) supplied by many computer languages. Each subroutine can be tested separately and then used in various programs. Subroutines can handle applications that frequently recur, such as reading or plotting data or carrying out a mathematical operation. This approach saves the time needed to write and debug portions of a program similar to one already available. Moreover, the overall structure of a program containing a set of calls to subroutines is generally easier to understand, because many complexities are isolated into subroutines.

- *Make programs comprehensible.* It is helpful to be able to understand programs once written. Clear documentation and modular programming help. In addition, it should be easy to tell what portions will be executed under which circumstances. For this purpose, portions of a program should be executed sequentially, rather than jumping backwards and forwards within a program.

Similarly, the statements themselves can be written clearly. The use of mnemonic variable names and natural groupings of variables can help. For example, it is somewhat unclear that

```
X = 0.23873*A / (Y*Y*Y)
```

gives the average density X of a planet with mass A and radius Y, whereas

```
RHO = AMASS / ((4.0/3.0) * PI * (RADIUS**3))
```

[2] Kernighan and Plauger (1978).

[3] Brooks (1975).

is clearer. For clarity, the latter expression is more verbose than required, has π previously defined, and is slightly less efficient.

- *Don't be clever.* Sometimes the shortest, "cleverest" way of programming something can be the worst. In addition to giving rise to lack of clarity, some shortcuts make it difficult to transfer programs between computers. This is especially true of programs that exploit specific properties of an individual computer or compiler, such as local variants of a standard programming language.

- *Keep a perspective on precision.* The program calculates and manipulates numbers that, at least in theory, correspond to physical entities. It is worth keeping track of the precision associated with the data and other quantities, and of that required to compute the desired results.

- *Organize programs and data.* Related programs and the associated files can be grouped into directories which include files listing and explaining the directory's contents. Data files can be organized similarly. Often seismograms, for example, go through multiple processing stages carried out by different programs. A common practice is to use specific types of file names to indicate various intermediate stages. In addition, the data files begin with *headers*, information identifying the data and recording the operations applied to it. The headers and file names should be updated by the programs themselves, rather than "by hand" at each stage. The output, whether text or graphic, should contain the parameters required to replicate the result. This can be especially important for interactive data processing because input files are not kept.

A.8.3 Representation of numbers

Several simple concepts about numerical calculations on a computer are worth bearing in mind. One is the consequences of the way in which numbers are represented and manipulated. Because computers use binary (base 2) arithmetic, numbers are written as sets of *bits*, single binary digits, grouped into *words*. Some general ideas about these representations can be illustrated without going into the schemes used by various computers.

Integers are represented by their binary equivalent. Thus 46 (decimal) is 101110, because

$$46 = 1\times 2^5 + 0\times 2^4 + 1\times 2^3 + 1\times 2^2 + 1\times 2^1 + 0\times 2^0.$$

Many computers represent integers by 16- or 32-bit words. The word length governs the range of possible integers. For example, using 16 bits, one of which indicates the sign, the largest positive integer that can be represented is

$$111\ 1111\ 1111\ 1111 \text{ (binary)} = 2^{15} - 1 = 32{,}767.$$

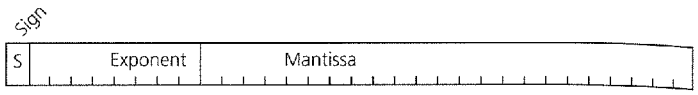


Fig. A.8-4 Representation of a floating point number using 32 bits.

Because a greater range is needed for scientific computation, floating point numbers are used:

$$\text{number} = (\text{mantissa}) \times 2^{\text{exponent}}.$$

Floating point numbers can accommodate fractions, with digits to the right of the binary point representing negative powers of two, just as digits to the left of the point represent positive powers of two. For example,

$$\begin{aligned} 46.625 \text{ (decimal)} &= 1\times 2^5 + 0\times 2^4 + 1\times 2^3 + 1\times 2^2 + 1\times 2^1 \\ &\quad + 0\times 2^0 + 1\times 2^{-1} + 0\times 2^{-2} + 1\times 2^{-3} \\ &= 101{,}110.101 \text{ (binary)} = 0.101110101\times 2^6. \end{aligned}$$

To represent binary floating point numbers on a computer, a certain number of bits are assigned to the mantissa and the exponent. Figure A.8-4 shows one way in which a single precision floating point number might be represented by a 32-bit word. One bit is reserved for the sign of the mantissa, 8 bits are used for the exponent including its sign, and the remaining 23 bits contain the mantissa. The number of bits available for the exponent determines the *range* of the floating point numbers. Because $2^8 = 256$, the exponent can represent numbers between approximately 2^{127} and 2^{-128} or approximately 10^{38} to 10^{-39}. The number of bits in the mantissa determines the *precision* or number of significant digits. Because 2^{-23} is approximately 10^{-7}, the maximum number of significant decimal digits is about seven. Further precision can be obtained using *double precision* numbers with additional bits for the mantissa. The precise values of the range and the precision depend on details of the implementation.

The range and precision in use are worth bearing in mind because computers do not always issue "overflow" or "underflow" warnings. The computer may assign a value, such as the largest floating point number, and proceed. It can be frustrating to find that the peculiar answers produced by a program result from numbers outside the computer's range.

A related malady is *round-off error*, the loss of computational precision due to the limited number of significant digits. To illustrate the concept, suppose that a computer used six bits for the mantissa. The decimal addition

$$0.65625 + 0.96875 = 1.625$$

would, in binary, be

$$0.10101 + 0.11111 = 1.10100,$$

which, because no precision was lost, equals the exact answer. Now, consider the decimal addition

$$5.25 + 0.96875 = 6.21875,$$

which, in binary, becomes

$$0.101010 \times 2^3 + 0.111110 \times 2^0.$$

To carry out the binary addition, because the numbers have different exponents, the mantissa of the smaller number is shifted to produce a common exponent. If some of the bits representing the smaller number are lost, inaccuracy may result. For example, in this case,

$$0.101010 \times 2^3 + 0.000111 \times 2^3 = 0.110001 \times 2^3$$
$$= 6.125 \text{ (decimal)}.$$

The precision available on a computer is generally adequate to avoid significant round-off error. Nonetheless, it is a potential problem to keep in mind, especially in long calculations or in those such as a series sum where the answer is the difference between large numbers.

A.8.4 A few pitfalls

Difficulties often can be avoided by considering how various statements in the program will be executed. This is especially the case when using compilers that provide little error checking and few helpful warning and error messages. The computer, following its explicit rules, may yield results differing from those expected. The foibles here are for Fortran, but similar ones often appear in other computer languages.

- *Statement execution.* Problems often stem from the distinction between integers and floating point numbers. For example, if I and J are integer variables,

```
J = 5
I = 1/J
```

yields zero, because integer division yields an integer. This problem is not cured by setting the result equal to a floating point variable, or performing a floating point operation on the integer result:

```
X = 1/J
Z = 1.0*(1/J)
```

yield zero, because division is done as an integer operation, and the result (0) is converted to floating point (0.0). On the other hand, most compilers give 0.2 as the result of

```
X = 1.0/J,
```

although a conservative policy is to explicitly convert the integer to floating point

```
X = 1.0/FLOAT(J).
```

A second class of problems can result from the order in which operations are performed. For example, it may be unclear whether

```
-1.0**2
```

should be interpreted as $(-1.0)^2 = 1.0$ or $-(1.0)^2 = -1.0$. Although the computer language rules are explicit, it may be wise to use parentheses, e.g.,

```
(-1.0)**2
```

to ensure that operations are carried out as desired. The additional parentheses can also make the program more comprehensible.

- *Subroutines.* Subroutines are heavily used in writing scientific programs. As a result, problems can result while using computer languages like Fortran in which what appear to be arguments passed to a subroutine are actually the locations in memory of these arguments.

A common error is exemplified by the following program

```
CALL SUB(1.0)
X = 1.0
WRITE (6,*) 'X = ', X
STOP
END

SUBROUTINE SUB(Y)
Y = 5.0
RETURN
END
```

which, when executed, yields "X = 5.0." Because Y, a parameter in the subroutine definition, was set equal to 5.0, the value of the corresponding parameter in the subroutine call, "1.0" has been redefined as 5.0. This situation, which sometimes underlies inexplicable behavior by programs, can be avoided by not passing numerical values of an argument explicitly to a subroutine if the argument will be redefined. For example, had the first statements been

```
Z = 1.0
CALL SUB(Z)
```

the variable Z would equal 5.0, but "1.0" would not be affected.

Other errors occur when either the type or number of arguments in a call to a subroutine do not match those in its definition. For example, calling a subroutine with an integer variable may yield unexpected results if the definition is in terms of a real variable.

• *Arrays.* Scientific computing often involves dealing with *arrays*, groups of data addressed by their indices. For example, a seismogram giving a single component (e.g., vertical) of ground motion can be written as an array (U(1), U(2) . . .) of displacement versus time. Similarly, a seismogram giving all three (vertical, north–south, east–west) ground motion components can be written as a two-dimensional array

```
U(1, 1), U(1, 2), U(1, 3), U(1, 4) . . .
U(2, 1), U(2, 2), U(2, 3), U(2, 4) . . .
U(3, 1), U(3, 2), U(3, 3), U(3, 4) . . .
```

whose first index gives the component, and second index indicates the time.

Arrays are defined initially by statements giving their dimensions, i.e.,

```
DIMENSION A(N, M)
```

or

```
REAL A(N, M).
```

Typically, the computer selects a memory location for the first element in A and reserves N × M successive locations. Similarly, N × M × R locations are reserved for a three-dimensional array dimensioned (N, M, R). In Fortran, regardless of the number of dimensions, an array is stored as one-dimensional with the first index varying the most rapidly, then the second, and so on. In other words, if A is dimensioned (2, 3), the storage order is

```
A(1,1), A(2,1), A(1,2), A(2,2), A(1,3), A(2,3).
```

For two-dimensional arrays, this can be thought of as storing the array by columns. An individual array element is found by calculating its location relative to that of the first element. Thus, for an array dimensioned (N, M), with element (1, 1) at location 1, element (I, J) is found at location

```
1 + (I - 1) + (J - 1) × N.
```

Several computational difficulties can arise in dealing with arrays. A common set of errors involve being "off by one," either by starting or ending on the wrong element. This is especially easy because some computer languages (e.g., Fortran) start with the first element in an array being "1," whereas others (e.g., "C") start with the first array element as "0." Thus one needs to make sure that the array elements correspond to the expected variable values, such as seismic record times. Often, when an array index is computed by the program, an error yields an index outside the bounds dimensioned for the array. Because many compilers do not check for such errors unless specifically requested, a statement like

```
A(9) = 4.0
```

will usually be executed even for an array dimensioned

```
DIMENSION A(5).
```

Typically, the computer places 4.0 in whatever is 8 locations in memory beyond A(1). This location may contain some other variable, or a portion of the program itself. Often the program continues until it requires the contents of the overwritten location, at which point several things may occur. At best, the program "crashes"; at worst, it continues the calculation with erroneous values that propagate. Array element out-of-bounds problems are among the most common and most frustrating difficulties in scientific programming. When a compiler provides array bounds checking, it is worth using.

The nature of array storage can also lead to inefficient programs. On many computers, data which are actually on disk can be treated as resident in memory, and are automatically "swapped" into physical memory when needed. For efficiency, large adjacent regions of the disk are often swapped into physical memory together. Efficient programs minimize swapping by making the most possible use of data that reside in physical memory. By contrast, inefficient programs can produce "thrashing," a situation in which much of the computer's time is spent swapping rather than computing.

For example, consider[4]

```
DIMENSION A(1000, 1000)
DO 10 I = 1,1000
DO 10 J = 1,1000
10 A(I, J) = I + J
```

Because the elements of A are stored in column order, A(1, 1) and A(1, 2) are a thousand locations apart. It would be more efficient to reverse the loops

```
10 A(J, I) = I + J
```

so that adjacent locations (A(1, 1), A(2, 1) . . .) were used successively.

• *Uninitialized variables.* Problems frequently result from *uninitialized variables*: those used in calculation without their values being set. A common example, summing an array

```
DO 10 I = 1, N
10 SUM = SUM + A(I)
```

can give strange results unless the compiler initializes SUM as zero. Because this is not always the case, it is thus wise to explicitly initialize, e.g.,

```
SUM = 0.0
```

before executing the loop. Proper initialization also helps to ensure that programs do not give different results on different computers.

• *The computer may be wrong.* Although most problems result from programming errors, a very small fraction of the time the error may be the computer's. Compilers have been known to contain "bugs" in common routines such as square root, tangent, or complex arithmetic. This tempting explanation for the failure of a long and intricate program can generally be rejected unless a test program that carries out only the suspect operation yields the wrong answer.

A.8.5 Some philosophical points

To close our discussion, a few general thoughts are worth considering. Historically, computers were considered a scarce and valuable resource. Currently, as computer power increases and costs fall, it is increasingly practical to carry out investi-

[4] Hatton (1983c).

gations numerically. One example is the change, both in exploration and in global seismology, from earth models whose properties vary only with depth, to three-dimensional models that are evaluated numerically.

The role of analytic solutions is also changing. In addition to the traditional goal of providing exact solutions to simplified problems, analytic solutions provide test cases for numerical solutions of more complex problems. Analytic solutions can also yield the insight needed to evaluate numerical results.

Along with the increase in the complexity of problems that can be solved computationally comes an increase in the volume of output. Fortunately, a parallel development has been the increasing role of graphic output, often in color. The proverb "A picture is worth a thousand words" may be unduly conservative in this context. A thousand words on a computer might be 32,000 bits; graphic output often makes it possible to visualize data with millions of bits.

Finally, software such as spreadsheets or programs with sophisticated general mathematical capabilities often eliminates the need to write programs for a specific application. In this book, we do not assume that such software will be used for the problems, although many could be done this way. We think that programming without using such software gives a deeper understanding of the underlying principles. Hence, in educational applications, we strongly favor programming, even if in non-educational applications ease of use may favor sophisticated software.

Further reading

Many texts cover portions of the mathematical material summarized here. Feynman (1982) discusses general issues of the relations between mathematics and science. Butkov (1968) and Menke and Abbott (1990) provide introductions to many of these topics. Fung (1969), Hay (1953), Jeffreys and Jeffreys (1950), and Marion (1970) treat vectors, vector transformations, and vector differential operators. Applied linear algebra texts such as Franklin (1968) and Noble (1969) deal with the range of the subject including numerical methods.

Articles by Hatton (1983a–d, 1984a,b, 1985) provide a broad and witty introduction to computer science for geophysicists. Eckhouse and Morris (1979) and Sloan (1980) cover topics in computer software, including the representation of numbers and arithmetic operations. Kernighan and Plauger (1976, 1978) discuss topics in programming style. Brooks (1975) treats issues in the development and organization of computer software. Numerical analysis texts like Froberg (1969) cover round-off and other sources of error in numerical computations. Harkrider (1988) gives an entertaining anecdotal account of early (*c.*1960) computer usage in seismology.

The application of spherical geometry to the paths between an earthquake and a receiver, including the effects of the earth's ellipticity are discussed by Ben-Menahem and Singh (1981) and Bullen and Bolt (1985). The theory of the earth's shape is treated by Cook (1973) and Jeffreys (1976).

Problems

1. Find the angle between the vectors (1, 4, 2) and (2, 3, 1).
2. Show, using index notation, that for the three-dimensional vectors **a**, **b**, **c**:
   (a) $\mathbf{a}\times\mathbf{b}$ is perpendicular to both **a** and **b**.
   (b) $|\mathbf{a}\times\mathbf{b}| = |\mathbf{a}||\mathbf{b}|\sin\theta$, where θ is the angle between the two vectors.
   (c) $\mathbf{a}\cdot(\mathbf{b}+\mathbf{c}) = \mathbf{a}\cdot\mathbf{b}+\mathbf{a}\cdot\mathbf{c}$.
   (d) $\mathbf{a}\times(\mathbf{b}+\mathbf{c}) = \mathbf{a}\times\mathbf{b}+\mathbf{a}\times\mathbf{c}$.
   (e) $\mathbf{a}\cdot(\mathbf{b}\times\mathbf{c}) = \mathbf{b}\cdot(\mathbf{c}\times\mathbf{a}) = \mathbf{c}\cdot(\mathbf{a}\times\mathbf{b})$.
   (f) $\mathbf{a}\times(\mathbf{b}\times\mathbf{c}) = \mathbf{b}(\mathbf{a}\cdot\mathbf{c}) - \mathbf{c}(\mathbf{a}\cdot\mathbf{b})$.
3. Show that for arbitrary matrices A, B, and C:
   (a) $(AB)^T = B^TA^T$.
   (b) $(ABC)^T = C^TB^TA^T$.
4. Prove the following properties of determinants for the case of a 2×2 matrix:
   (a) The determinant of a matrix equals the determinant of its transpose.
   (b) If two rows or columns of a matrix are interchanged, the determinant has the same absolute value, but its sign changes.
   (c) If a multiple of one row (or column) of a matrix is added to another row (or column), the determinant is unchanged.
   (d) If two rows or columns of a matrix are the same, the determinant is zero.
5. Express the determinant of a 3×3 matrix using the definition in Eqn A.4.17.
6. Prove that if A has an inverse, the two solutions **x** and **y** satisfying $A\mathbf{x} = \mathbf{b}$ and $A\mathbf{y} = \mathbf{b}$ are equal.
7. Find the inverse of the matrix
$$\begin{pmatrix} 1 & 2 \\ 5 & 4 \end{pmatrix}$$
   both by the cofactor method and by row operations. Check that the solution is in fact the inverse.
8. Show that the inverse of a 2×2 matrix A is given by
$$A^{-1} = \frac{1}{|A|}\begin{pmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{pmatrix}.$$
9. Show that A, the transformation matrix for a rotation about the $\hat{\mathbf{e}}_3$ axis (Eqn A.5.9) satisfies $A^TA = I$ and is thus orthogonal.
10. Prove that the magnitude of a vector is preserved by an orthogonal transformation.
11. Expand the determinant that give the eigenvalues of a 3×3 matrix (Eqn A.5.19) and verify that the invariants (Eqn A.5.21) are the coefficients of the characteristic polynomial.
12. Prove the following vector identities using index notation:
    (a) For any vector field $\mathbf{u}(\mathbf{x})$, $\nabla\cdot(\nabla\times\mathbf{u}) = 0$.
    (b) For any scalar function $\phi(\mathbf{x})$, $\nabla\times\nabla\phi = 0$.
13. For the vector field $\mathbf{u}(x, y, z) = (3x^2y^2 + z, 2x^3y + 2y, x)$, find:
    (a) $\nabla\cdot\mathbf{u}$.
    (b) $\nabla\times\mathbf{u}$.
    (c) $\nabla^2\mathbf{u}$.
    (d) A scalar field $\phi(x, y, z)$ such that $\mathbf{u} = \nabla\phi$.

14. Use index notation to show that the Laplacian in Cartesian coordinates of any vector field $\mathbf{u}(\mathbf{x})$ satisfies

$$\nabla^2\mathbf{u} = \nabla(\nabla\cdot\mathbf{u}) - \nabla\times\nabla\times\mathbf{u}.$$

15. Show that at any point in a spherical coordinate system, the spherical basis vectors $(\hat{\mathbf{e}}_r, \hat{\mathbf{e}}_\theta, \hat{\mathbf{e}}_\phi)$ form an orthonormal set.
16. Use Eqn A.7.6 to derive the angular distance Δ between the locations of an earthquake and a seismic station as given in Eqn A.7.7.

Computer problems

The solutions may be useful for other problems in this and other chapters.

C-1. Find the largest integer your computer allows by starting with "2," "2 × 2," "2 × 2 × 2," and doing successive multiplication by 2. What happens when you exceed this number? Do the same for floating point numbers using "10.0" instead of "2" in both single and double precision. Does double precision allow larger floating point numbers?

C-2. Find when your computer starts to show round-off error by starting with "10.0" and doing successive multiplications by 10.0. At each step, add 1.0 to the result and subtract the two numbers. When does the difference become zero? Do the same in double precision.

C-3. Write subroutines to do the following operations on an input vector in three dimensions:
(a) Find the magnitude of a vector.
(b) Find the sum of two vectors.
(c) Find the scalar product of two vectors.
(d) Find the vector product of two vectors.
Your subroutines should include comment lines explaining the purpose of the routine and the various inputs and outputs.

C-4. Write a subroutine using the necessary subroutines from problem C-3 to find the angle between two vectors.

C-5. Use the solutions to problems C-3 and C-4 to find the magnitude, sum, scalar product, and vector product of the vectors (1, 4, 2) and (2, 3, 1), and the angle between the two vectors.

C-6. (a) Write a subroutine to multiply an $n \times m$ matrix by an m-element vector.
(b) Write a subroutine to multiply an $n \times m$ matrix by an $m \times r$ matrix.
(c) Write a subroutine to find the determinant of a 3×3 matrix.

C-7. (a) Write a subroutine that uses Gaussian elimination with partial pivoting to solve the system of equations $A\mathbf{x} = \mathbf{b}$. The routine should take an arbitrary 3×3 matrix A and 3-element vector $\mathbf{b}$ as inputs. The program should test the solution by multiplying $A\mathbf{x}$ and subtracting $\mathbf{b}$ from the result. The subroutines from C-6 may be helpful.
(b) Use the subroutine to solve

$$\begin{pmatrix} 10 & -7 & 0 \\ -3 & 2 & 6 \\ 5 & -1 & 5 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 7 \\ 4 \\ 6 \end{pmatrix}.$$

C-8. (a) Write functions that return the values of the δ_{ij} and ε_{ijk} symbols given the indices as arguments. Test the functions and show that they give the correct values.
(b) Write a program that uses these two functions to prove the identity

$$\varepsilon_{ijk}\varepsilon_{ist} = \delta_{js}\delta_{kt} - \delta_{jt}\delta_{ks}$$

by testing all possible combinations of indices.

C-9. (a) Write a subroutine to invert a 3×3 matrix using elementary row operations. The subroutine should first check to see if the matrix is singular. It should test the result by multiplying by the original matrix.
(b) Use this routine to invert

$$\begin{pmatrix} 1 & -1 & -1 \\ 3 & -1 & 2 \\ 2 & 2 & 3 \end{pmatrix}.$$

C-10. (a) Write a program to solve a 3×3 system of equations $A\mathbf{x} = \mathbf{b}$ using the matrix inversion routine from the previous problem. The program should test the solution by multiplying $A\mathbf{x}$ and subtracting $\mathbf{b}$ from the result. The subroutines from C-6 may be helpful.
(b) Use the program to solve the system of equations in C-7.

C-11. (a) Write a subroutine to find the roots of a general cubic equation using the method given below.[1]

A cubic equation $y^3 + py^2 + qy + r = 0$ may be converted to

$$x^3 + ax + b = 0$$

by defining

$$y = x - p/3, \quad a = (3q - p^2)/3, \quad b = (2p^3 - 9pq + 27r)/27.$$

If p, q, and r are real, the quantity

$$c = b^2/4 + a^3/27$$

characterizes the roots: if $c > 0$, there is one real root and two conjugate imaginary roots; if $c = 0$, there are three real roots, of which two are equal; and if $c < 0$, there are three real and unequal roots. Using

$$A = (-b/2 + c^{1/2})^{1/3}, \quad B = (-b/2 - c^{1/2})^{1/3},$$

the values of x given by

$$x = A + B, \quad [-(A+B) + (A-B)\sqrt{-3}]/2,$$
$$-[(A+B) + (A-B)\sqrt{-3}]/2$$

are the roots.

The subroutine requires complex arithmetic and should test the roots by substituting back into the equation.

(b) Use the result to solve

$$y^3 - 8y^2 + 19y - 12 = 0.$$

C-12. (a) Write a subroutine to find the eigenvalues and eigenvectors of a real, symmetric 3×3 matrix, using the results of C-11. The program should check that the eigenvectors and eigenvalues satisfy their definition. Be careful to avoid dividing by zero.
(b) Use this subroutine to find the eigenvalues and eigenvectors of

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{pmatrix}.$$

[1] Beyer (1984).

C-13. (a) Write a program that accepts the latitude and longitude of two points on the earth's surface and finds the angular distance and distance along the earth's surface between them, and the azimuth and back azimuth.

(b) Use your program to find the distances and azimuths between:

(i) Cairo, Illinois (37°N, 89°W) and Cairo, Egypt (30°N, 32°E).

(ii) Berlin, New Hampshire (44.5°N, 71.5°W) and Berlin, Germany (52.5°N, 13.5°E).

(iii) Montevideo, Minnesota (45°N, 95.5°W) and Montevideo, Uruguay (35°S, 56°W).

(iv) Mexico, Maine (44.5°N, 70.5°W) and Mexico City, Mexico (19°N, 99°W).